Quantifying the Overall Impact of Caching and Replication in the Web

M. Baentsch, A. Lauer, L. Baum, G. Molter, S. Rothkugel, P. Sturm
{baentsch, lauer, lbaum, molter, sroth, sturm}@informatik.uni-kl.de

02 / 1997

Sonderforschungsbereich 501

AG Systemsoftware
Computer Science Department
University of Kaiserslautern
P.O. Box 3049, D-67653 Kaiserslautern

Quantifying the Overall Impact of Caching and Replication in the Web

M. Baentsch, A. Lauer, L. Baum, G. Molter, S. Rothkugel, P. Sturm

AG Systemsoftware, Computer Science Department,

University of Kaiserslautern, P.O. Box 3049, D-67653 Kaiserslautern

Email: {baentsch, lauer, lbaum, molter, sroth, sturm}@informatik.uni-kl.de

Abstract

This paper discusses the benefits and drawbacks of caching and replication strategies in the WWW with respect to the Internet infrastructure. Bandwidth consumption, latency, and overall error rates are considered to be most important from a network point of view. The dependencies of these values with input parameters like degree of replication, document popularity, actual cache hit rates, and error rates are highlighted. In order to determine the influence of different caching and replication strategies on the behavior of a single proxy server with respect to these values, trace-based simulations are used. Since the overall effects of such strategies can hardly be decided with this approach alone, a mathematical model has been developed to deal with their influence on the network as a whole. Together, this two-tiered approach permits us to propose quantitative assessments on the influence different caching and replication proposals (are going to) have on the Internet infrastructure.

1. Introduction

Caching and replication are proven techniques for scaling up most computing or communications infrastructures [HK+88]. With the rapid deployment of the World-Wide Web the use of caching infrastructures quickly became a necessity. Recently, following the example of filesystem design, many different concepts for data replication in the Web have been developed. All of these approaches are aimed at solving only singular problems experienced by both users and administrators of the WWW. For example, in order to reap the benefits with respect to server load reduction, increased document availability, and latency reduction, one has to sacrifice bandwidth. Due to the contradictory nature of the involved variables the concentration on one particular set of parameters in caching and replication systems is an inevitable limitation. However, so far no work has been presented that makes explicit these trade-offs from both perspectives of a single cache and the overall network infrastructure.

In particular, this paper aims at providing answers to questions like: What percentage of WWW requests made by clients using a caching hierarchy have to be answered by the origin servers? How much network traffic is generated by WWW requests at which locations? To what extent does a caching infrastructure influence the failure behavior of the WWW? In what way do these numbers change if a particular replication concept is used? Where are the most profitable locations for replica servers?

In order to provide answers to these issues, two complementary approaches are followed in this work. The first one is a trace-based simulation of the behavior of a WWW cache given different caching and replication parameters, algorithms, and heuristics. Derived from the results and experiences gained during the use of this system, a more general model of the network entities involved in caching and replication strategies for the Web has been developed. This two-tiered approach is backed by the evaluation of 2.7 million log entries of our university's main proxy and over 100 million logfile entries of major backbone caches.

Thus, this paper is structured as follows. The ensuing section contains an overview on related research in the area of development and evaluation of caching and replication techniques for the WWW. A brief introduction to methodology and prerequisites for our work is given in section 3. Section 4 contains an overview on the design and implementation of a simulator for a caching proxy enhanced with replication facilities. The structure and information of the traces forming the base of this work is also explained in this section. In section 5, a mathematical model for the overall effect of caching and replication techniques in the WWW is suggested. Assumptions, necessary simplifications, and initial results are reported on. A comparative analysis of the two approaches with a distinct focus on data replication is given in section 6 before the paper concludes.

2. Related Research

The advantages of caching within the Internet have been discussed for example in [DHS93]. In particular, the influence of caches within the backbone infrastructure of the Internet has been examined for FTP traffic. One result of that work was the development of the Harvest system and its caching proxy for the WWW [CD+96]. The resulting software over time started to boast more installations than the original WWW caching proxy distributed by CERN/the WWW Consortium [LuA94]. Especially from the performance point of view, the Harvest proxy, resp. systems as the one presented in [Wes95] clearly have the leading edge. Other approaches are aimed at improving the performance of caches by introducing further parameters significant for the caching algorithms, e.g. user behavior [PiR94] or document properties [WA+96].

A distinctive separation between caching and replication proposals for the World-Wide Web is hardly visible. For example, proposals like geographical push-caching [GwS95], or speculative service [Be95a] may be considered initial approaches to introducing active replication into the Web. More technology-driven proposals are given in [Don95], [MLB95], or [Be95b]. In general, these proposals require rather drastic modifications to the basic HTTP protocol, whereas other researchers' proposals remain compatible with current Web technology.

Three approaches falling in the area of replication-based proposals can be differentiated. The first one proposes to preload existing caches primarily in order to reduce cache staleness (e.g. [WaC96]). A second one takes the opposite position and calls for server-initiated cache invalidation, resp. active data replication (e.g. [Wes95], [GwS95], [Be95b]). An approach trying to find a middle ground in both issues argues for replica servers whose establishment is controlled by traffic information collected by proxies. In combination with a simple, application-level naming service it also permits to transparently redirect clients to appropriate replica servers [BMS96]. Research results as [CPB93] confirming the existence of high traffic favoritism also make the point that the latter approach might be best suited to distribute the most often requested documents farthest into the net.

As far as simulations of caching behavior in the WWW are concerned, several —in part contradictory— findings are known. For example, [CD+96] does not advocate active invalidation strategies, whereas [GwS96] argue in favor of it. In any case, the importance of simulations relying on real-world data has been pointed out explicitly in those presentations. Finally, the different works surveyed tend to concentrate on many, but not all aspects at stake when considering the influence of caching and replication techniques on the Web. Some trade latency for bandwidth [GwS95], others improve cache hit rate [AS+95]. Yet other research suggests that a concentration on byte hit rate, i.e., the actual amount of data conserved by employing caching, is more important [BC+95]. Server load reduction [Be95a] as well as increased failure tolerance as visible to users of the WWW also form the base of further research [BMS96].

As far as modeling concepts for internetwork structures are concerned, many proposals are known, like [ZCB96], or [VNo92]. However, these approaches are in general very much focused on researching particular aspects of packet-oriented networking and to a lesser degree deal with the application-level perspective. Therefore, this work aims at assessing end-to-end application performance as opposed to end-to-end packet transmission performance as considered for example in [Bol93]. Three basic reasons caused us to select this approach. First, reliability and performance of the —application-level— proxy software has to be considered. Second, our observations from evaluations of different kinds of proxy logfiles affirm the assumption that the law of large numbers applies in this area of measuring bandwidth, latency, and error rate. In our opinion, this obviates the need to consider the complex network behavior at packet level. Finally, this approach permits us to concentrate on the effects of the HTTP protocol without having to consider network-level phenomena.

3. Foundations

This section first discusses the basic methodology employed for tackling the questions at hand. Secondly, the raw data used for our approach are introduced.

Methodology

The foundation for both approaches discussed in this paper are traces obtained from real-world proxy servers. Moreover, we have modified our university’s main WWW proxy to log additional information of importance to more accurate simulations. This server has been run for a period of six months starting in January of 1996.

During this time, 2.7 million requests have been served, giving our simulator an acceptable broad data set. For example, the logs have been evaluated to measure bandwidth and latency at application-level for sites from which documents have been requested. This approach also remedies the problem of erratic behavior with respect to measurable latency and loss rates in case of probe traffic making up only a small fraction of the overall traffic [Bol93]. Accordingly, our "probe" traffic consists of the actual application traffic and thus our measurements do not exhibit a random behavior — resulting in representative figures like fig. 1. Moreover, such passive log evaluations do not create any additional network overhead.

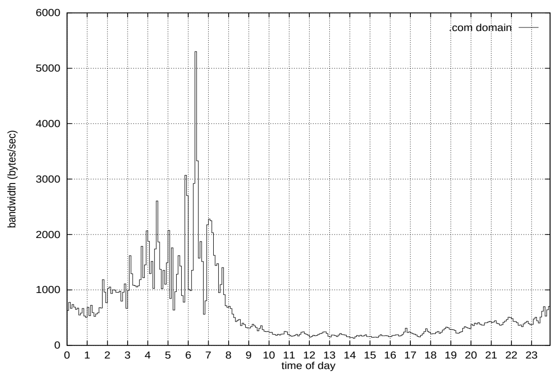


Figure 1. Average bandwidth as measured between our proxy and WWW servers in the .com domain.

This evaluation also provided input data to our model of the World-Wide Web as discussed in section 5. In particular, data access rates, cache hit rates, and document popularity have been measured. In order to not only rely on this comparatively local data set, we also used the logs of more globally operating proxies for obtaining input data to our overall network model. About 100 million accesses from the proxies in the Distributed Testbed for National Information Provisioning [NLANR] have been obtained and evaluated for this purpose.

Input data

Access patterns obtained from the logs of our university’s main proxy server reflect the traffic in a typical proxy server scenario (see fig. 2) where clients send their requests via a proxy server. For each request, the logfile contains important input data for the simulation, like the client´s HTTP request itself, the date and time of day when the retrieval took place, and the duration of a document download. Further information contained in each log entry are the size of the retrieved document as well as the origin server’s reply code indicating whether the access was successful or not. Additionally, we modified the server code to also log the last-modified time stamp of each remotely accessed page. This additional information makes it possible to approximate the average change rate of WWW objects and furthermore permits our cache simulator to evaluate various cache coherence protocols.

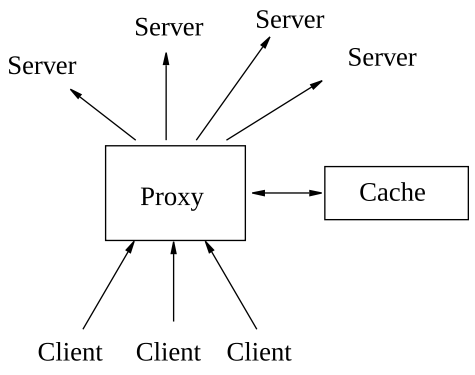


Figure 2. Proxy server scenario

4. Simulation

The first goal of this work was to learn in more detail about how different cache and replication parameters influence the performance of a single WWW proxy cache. To this end, we designed a cache simulator called *CaSimiR* (*Ca*che *Sim*ulator with *i*ntegrated *R*eplication strategies). It provides the basic environment for evaluating different cache configurations and for visualizing the results obtained by evaluating real-world WWW proxy logs. The presentation of this work [Lau96] concentrates on the discussion of the basic features of the simulator with respect to handling replicated documents. A brief overview of some of the results gained while testing usual and new caching strategies is also given. The results include information on average document retrieval latency, bandwidth used, as well as other cache statistics like the cache (byte) hit-rate or the stale rate. The latter value is of particular importance to any heuristically working cache, as it indicates how often old objects were retrieved from the cache instead of a more current version on the original server. .

A typical simulation run

The following paragraph gives a brief description about the basic computational steps during a simulation run, typically evaluating the requests logged within one particular month. The simulator first sets all parameters like cache size, maximum size of cachable objects, or cache coherence strategy. For example, CaSimiR currently supports a time-to-live (TTL) strategy based on the age of the cached document. This means that each object in the cache is assigned a time frame during which it is considered valid and accesses to this object are served from the cache. Accesses to objects after the end of their TTL result in conditional HTTP requests being sent to the appropriate server. These requests are dubbed if-modified-since (IMS) requests since they aim at determining whether a document has changed on the origin server after a particular date or not. The length of the validity period is determined by the object's last modification time multiplied by the TTL factor.

The integration of replication in CaSimiR's caching scheme is realized by setting apart a fraction of the cache memory to hold replicated documents. All objects stored in this area are considered up to date. Technically, this means that (simulated) conditional IMS requests never have to be issued for a document in this part of the cache. CaSimiR provides for the basic functionality to load any set of pages into this area, thus allowing any replication strategy to be tested. Currently the most heavily referenced documents of the preceding month become replicated pages.

The next step of simulation consists of preloading the cache. Starting with a cold cache would result in many cache misses at the start of the simulation and would therefore not reflect the performance of a real world cache. Warming up the cache is typically done with access patterns from a month directly preceeding the one used for the simulation. When this preload is finished, CaSimiR starts to continuously read entries from the logfile containing the access patterns mentioned above. All necessary information like accessed URL, last modified time stamp, and object size is passed to the respective caching algorithm under consideration. Thus, CaSimiR maintains a complete image of the (simulated) cache at any time. For example, the URLs of stored objects are available, such as to decide whether an access was a cache hit or a miss. In order to compute accurate performance results, like latency, bandwidth demands, or stale rate, more information has to be maintained. In case of a cache hit the expiry time stamp of the accessed object is checked to determine whether a further (simulated) conditional IMS request has to be issued to check the object's freshness. If the object has been modified, the access has to be handled like a normal cache miss. In this case, the ensuing access evaluation phase adds an additional penalty in transfer time and transferred bytes for this access. Cache misses normally lead to the replacement of one or more objects in the cache. In this case, the caching algorithm updates its cache list, marks the actual page stored and deletes the replaced pages. Accesses to replicated pages are unconditionally handled in the same way as normal cache hits.

After complying to the respective caching and replication strategy under consideration, the statistics module of CaSimiR evaluates the access. The first value calculated is the duration of the access. Overall —average— latency information is updated accordingly. Of course, this is only possible if the logfile trace contains enough (URL-specific) information about loading times throughout a typical day. This information makes it possible to estimate transmission times for which no actual data can be found in the proxy logfile. These estimations are necessary for example when additional accesses like the conditional IMS requests mentioned above have to be taken into account. To reiterate, CaSimiR has to rely on these estimations as the actual proxy from which the traces have been obtained might have used a completely different caching strategy. Accordingly, no actual data for one particular URL at one particular time of day might be present in the trace.

The calculation of the required overall bandwidth measured in bytes retrieved from all the WWW servers via the proxy server is rather straightforward. If a cache miss is encountered, the number of bytes equaling the page size is considered sent from a server and therefore increases the overall bandwidth used. In any case, the HTTP protocol overhead (response header information) is counted, even if no actual data is transferred, e.g. in case of empty IMS responses. Otherwise, in case of a cache hit —or the retrieval of a replicated page—, no externally measurable bandwidth usage is assumed. As already mentioned above, the measure of bandwidth usage in our case does not include any update activities for the replicated pages.

Results

The following graphs are highlighting some of the simulation results. The first one shows the influence of different caching strategies and overall cache size on observable hit rate and externally observable bandwidth requirements (fig. 3). Several caching algorithms have been experimented with. Serving as a baseline strategy, Random denotes the algorithm replacing some random document if space is needed in the cache. LRU is the well-known least recently used data replacement algorithm. TTL denotes the strategy known for example from the CERN caching proxy [LuA94] to replace those pages whose time to live has expired. In this experiment, the TTL factor is set to 10%, resp. 30%. The last strategy under consideration has been dubbed Least Caching Profit (LCP). This novel algorithm aims at removing those documents promising the least benefit with respect to loading latency. This means that those documents are deleted from the cache which are considered to be more quickly reloaded and less often requested from the original server than other documents. This decision rests upon the measured loading duration as seen in the proxy logs and the net benefits acquired by this page while it was staying in the cache. The most important conclusion drawn from fig. 3 is the lesson that the LCP algorithm which strives to reduce user-perceptible latency results in lower byte hit rates, and accordingly in a worse bandwidth utilization. In turn, fig. 4 shows that indeed overall latency is reduced while document staleness is comparable with a usual LRU strategy. In the last figure highlighting pure caching effects, fig. 5 shows the influence of different limitations on the size of documents permitted to enter the cache under the LRU strategy. Decreasing the upper limits on cached documents size, results in increased bandwidth requirements as well as increased cache hit rates.

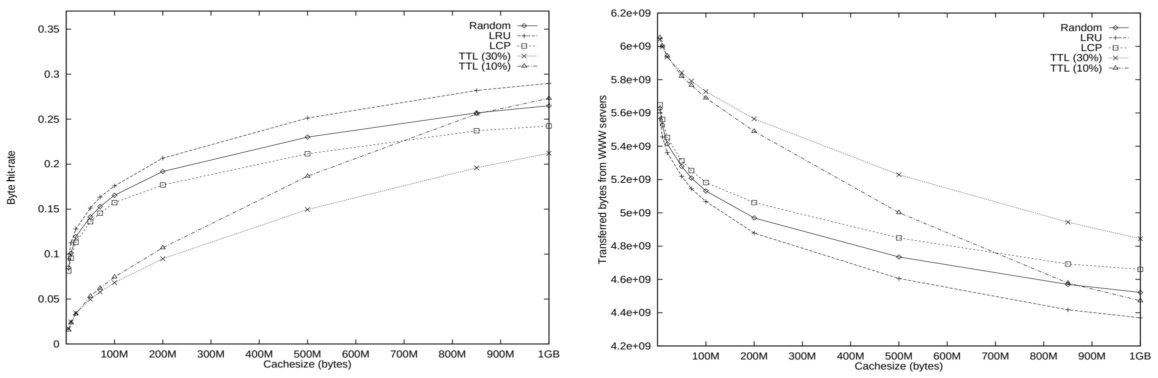


Figure 3. Byte hit rate and bandwidth as functions of cache size and caching algorithm used

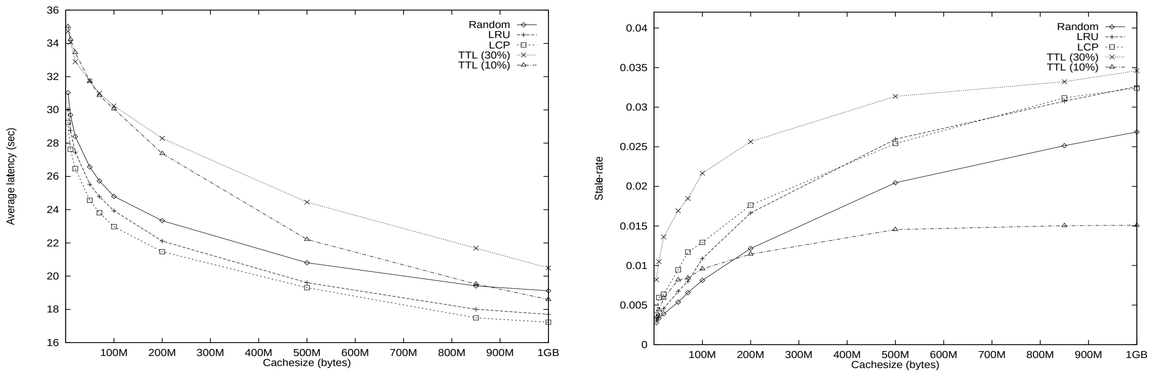


Figure 4. Latency and staleness as functions of cache size and different caching algorithms

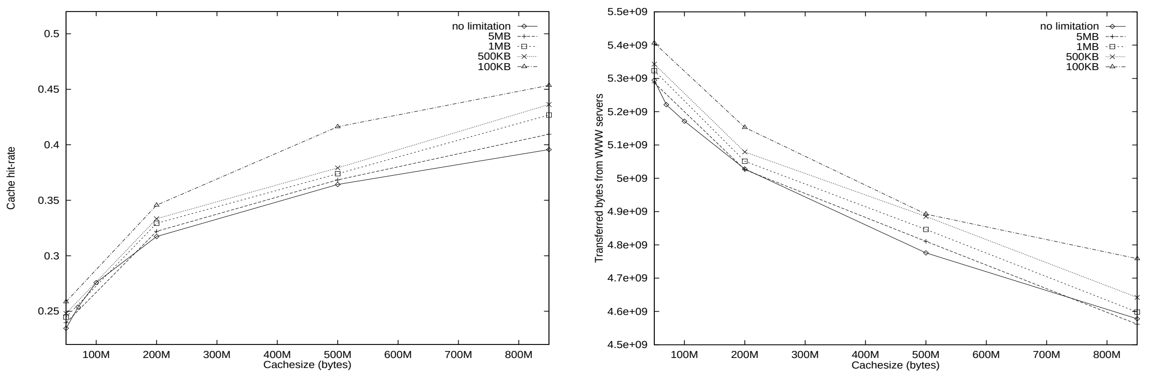


Figure 5. Cache hit rate and bandwidth as functions of cache size and minimum document size per cacheable object

One effect of setting aside a distinct part of the simulated caches' space for replicated documents is highlighted in fig. 6. Both values shown —cache hit rate and byte hit rate— represent the percentage of traffic served from the replicated area. One lesson drawn from this result is that bigger cache sizes —and accordingly larger replica space— does not have as much influence of the success of replication as the size of the reserved replication space. Finally, the most interesting results with regard to answering some of the questions brought up in the introduction are presented in fig. 7. The influence of different percentages of cache space set aside for replicated data is shown. Both values of interest —latency per request and overall transfer rate— are reduced. The latter of course only results from neglecting the amount of bandwidth required to replicate the data to the cache in the first place. Since this issue can only be tackled sensibly in a more general setting, it is one of the focal points of the next section.

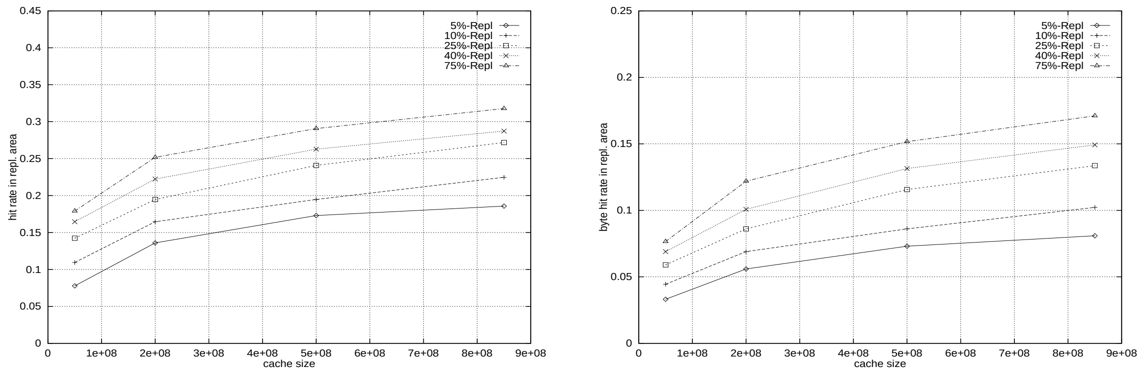
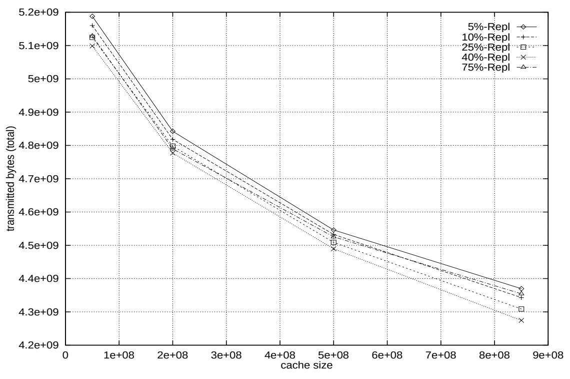


Figure 6. Percentage of all (byte) hits served from the replicated area of the simulated cache



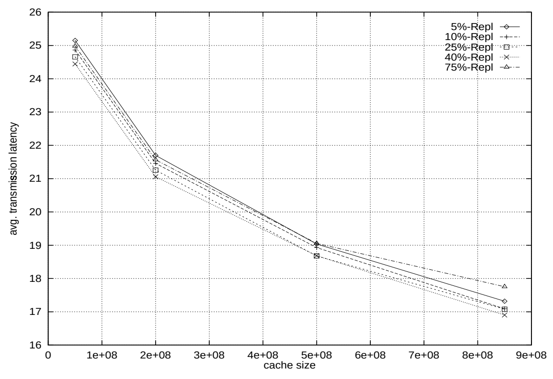


Figure 7. Bandwidth consumption and latency

5. A model of the WWW

The simulation results of the replication benefits in the WWW cannot be fully satisfying in two regards. On the one hand side, replication did not provide the sought after definitive improvements immediately visible at the level of a single proxy. On the other hand, some of the conceptual advantages of replication simply cannot be shown at this level. In order to quantify the influence of caching and replication on the overall network's reliability, latency, and bandwidth requirements, a more general model is needed. Thus, building on the experiences gained when designing the simulator, a model interconnecting the different parameters of a caching and replication system for the Web has been developed. Before introducing this more formal model, some prerequisites are in order. To start with, a list of simplifying assumptions about the modeled system as well as a set of definitions has to be introduced.

Assumptions

In order to strike a balance between simplifications and accuracy of the overall model, the system depicted in fig. 8 forms the base for the ensuing discussions.

This figure shows an application-level view of the client/server model of the WWW infrastructure using the Internet. WWW servers are located on the left hand side, proxies and clients on the right hand side. The cloud in the center represents the fault-tolerant backbone structure of the Internet. Closer to the clients and servers, the net degenerates into a hierarchic tree of connections whose failure would severe the respective sub-branch from the rest of the net. The rationale for this decision is given by both actual Internet service provider structures and research results indicating route stability for long periods of time [Pax96], resp. a small set of volatile routes [Chi93]. The dark dots within fig. 8 represent WWW proxies or —at the server side— WWW servers and accelerators [CD+96]. No further locations for intermediate servers are considered in this model. In particular, additional firewalls with caching functionality at any level on the server side of fig. 8 are not represented. The numbers and variables beneath the figure denote the distance in terms of network hops from a client. In this realm, d_a and d_e represent —average— entry and exit distance into the backbone mesh, whereas d_s denotes the —average— distance of a WWW server from its clients. Since we were not interested in determining the effects document replication has on the server side of the Web, only one branch of clients accessing the servers is printed in bold in fig. 8. An extension of the model to further branches of clients is considered for future work.

All in all, fig. 8 might be regarded in a more vivid, geographically oriented explanation as a representation of US WWW servers at the left hand side, and Asian or European proxies and clients at the right hand side.

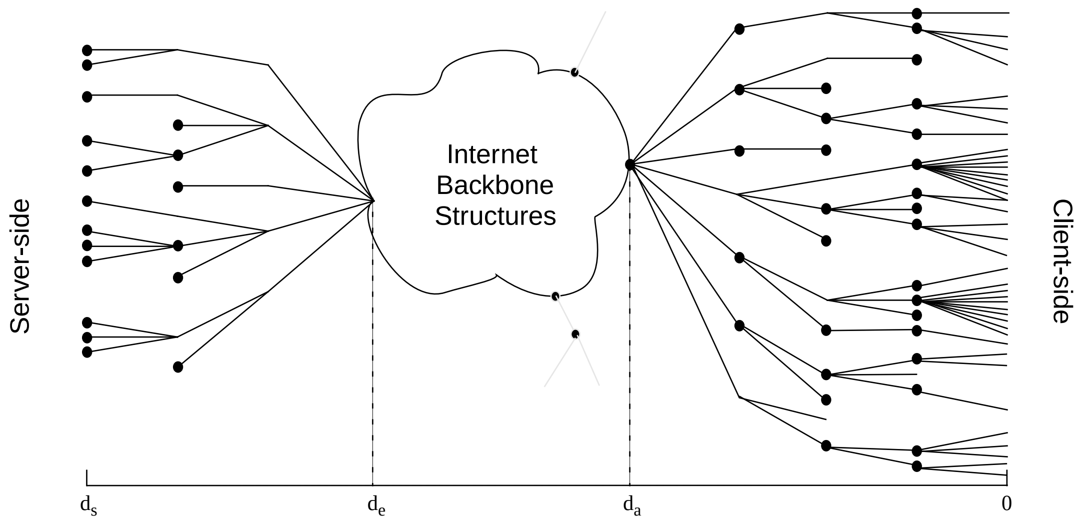


Figure 8. Overview on the application-level abstraction of the WWW underlying the presented model

Initial parameters

In order to determine the overall data flow with respect to bytes transferred and network hops traversed per request, the following parameters are introduced. Doc_{Size} denotes the average document size, Req_{Num} the average number of requests made by a single client. The percentage of conditional HTTP IMS requests per client is denoted by P_{IMS}. In contrast to P_{IMS} which determines the percentage of validation requests of —higher level caches— by lower level caches, $P_{Refresh}$ denotes the forced reloads of cache contents by the user via the respective browser button. The amount of server side accelerators, resp. firewalls with caching functionality is denoted by P_{Acc}. Parameters on the single network levels starting with 0 at the client site are differentiated by the subscript letter j. $Arity_j$ describes the number of immediate children of one particular node in fig. 8. Thus, $Arity_0$ accordingly is 1 as only one client is found at the end of this client-side tree. P_j designates the Byte-hitrate per level. At levels without a proxy cache ($d_a<j<d_{s-1}$), this number is set to 0. Accordingly, these levels are omitted from our model. In order to still correctly determine the amount of routers traversed by each request, the parameter $dist_r$ has been introduced. It is used to designate the —average— number of network hops between proxy servers at the receiver side of fig. 8. As far as the determination of overall system resiliency against faults is concerned, the parameter F_j has been introduced. It determines the daily failure probability of one single hop, i.e., router and proxy server (software) combined. The last parameter of importance to the explanation of the initial caching model is the probability for an IMS request to be satisfied at a particular level, Q_j. According to the semantics of this request, Q_{d_s} has to be 1 and $Q_j = 0$ for all levels j without a proxy ($d_a<j<d_{s-1}$).

Caching equations

Building on these initial definitions, the single levels are considered separately. On the recipient side, the percentages of data served *exactly at* any one particular level are denoted by S_j. S_j in turn recursively depends upon the percentage of requests served *up to* a particular level j (U_j). Serving in this context refers to the fact that a WWW request is satisfied by a cache —or later on, replica server— hit.

$$S_1 = U_1 = (1 - P_{Refresh}) \cdot (P_1 \cdot (1 - P_{IMS}) + Q_1 \cdot P_{IMS})$$

$S_j = (1 - P_{Refresh}) \cdot (1 - U_{j-1}) \cdot (P_j \cdot (1 - P_{IMS}) + Q_j \cdot P_{IMS})$ for all j with $1<j<=d_a$.

The first factor in this equation expresses the explicit disabling of caches via the no-cache pragma as issued by a browser in reaction to the —$P_{Refresh}$ percent— explicit document reload commands by the users.

$U_j = U_{j-1} + S_j$ for all j with $1<j<=d_a$.

The actual amount each level contributes to the overall average hop count per request can then be determined as $H_j = j \cdot (1 + dist_r) \cdot S_j$.

For all j with $d_a<j<d_{s-1}$, $S_j = 0$, as no proxies are located at those levels. Accordingly, the value of U_j does not change at these levels. The values for S and U only change at level d_{s-1} and d_s, i.e., at the accelerator/caching firewall and origin server respectively. The first one contributes

$S_{d_{s-1}} = (1 - U_{d_{s-2}}) \cdot P_{Acc} \cdot (P_{d_{s-1}} \cdot (1 - P_{IMS}) + Q_{d_{s-1}} \cdot P_{IMS})$ to $H_{d_{s-1}} = (d_a \cdot dist_r + 1) \cdot d_{s-1} \cdot S_{d_{s-1}}$.

Finally, the original server has to handle the requests not dealt with at any earlier level:

$S_{d_s} = (1 - U_{d_{s-1}})$ resulting in $H_{d_s} = (d_s + d_a \cdot dist_r)(1 - U_{d_s})$.

In summary, the overall average number of hops any single WWW request issued by a user has to travel is $H = \sum_{j=1}^{d_s} H_j$. This number can be also seen as the measurement for request latency. In order to fully equate H with the user-perceptible latency, an average dependency between per-hop latency, time of day, and number of hops has to be used. This has been shown to be hard to do on a network level due to high variances [SA+93]. Nevertheless, statistical statements based on average latencies for hosts more than a certain number of hops away are still considered to be possible (cf. fig. 1).

The two remaining values of interest in this model, i.e., overall bandwidth and failure rate, can be derived directly from the equations above. Overall bandwidth can be computed as the average number of hops per request (H) multiplied with the number of requests made by all clients within the spanning tree: $B = H \cdot \prod_{j=1}^{d_a} Arity_j \cdot Req_{Num} \cdot Doc_{Size}$.

For the failure rates per level the following equations hold:

$E_j = F_j \cdot \sum_{i=j}^{d_s} S_i$ for $j<d_{s-1}$.

$E_j = F_j \cdot (S_{d_{s-1}} + S_{d_s})$ for any level without a proxy ($d_a <j < d_{s-1}$).

$E_{d_{s-1}} = (S_{d_{s-1}} + S_{d_s}) \cdot (F_{d_{s-1}} + F_{d_s})$ and $E_{d_s} = S_{d_s} \cdot F_{d_s}$.

The overall failure rate results in $E = \sum_{j=1}^{d_s} E_j$.

In order to be able to apply these results to single requests, only a few considerations are necessary. First, E and H already denote per-request values by definition. B in turn has to be divided by the overall number of requests made by any client on the recipient side of fig. 8. Thus, the following equations hold:

$E_{URL} = E$, $H_{URL} = H$, and $B_{URL} = B \Big/ \left(\prod_{j=1}^{d_a} \mathrm{Arity}_j \cdot \mathrm{Req}_{Num} \right) = H \cdot \mathrm{Doc}_{Size}$.

Results with respect to caching behavior

In order to determine the validity of this model, most of the parameters given have to be filled in with concrete values. These have been gained from evaluating the proxy logfiles as described in section 3. Moreover, cache log evaluation results as reported in [BC+95], [Wes95] have been taken into consideration. In particular, the following numbers form the base of the ensuing presentation of results.

Table 1. Per-level parameters for evaluating the model

Level	**$Arity_j$**	**P_j**	**Q_j**	**F_j**	
0	1	40.0 %	0.0 %	0.01 %	Client-local cache
1	5	30.0 %	0.1 %	0.001 %	LAN level cache
2	10	20.0 %	0.5 %	0.0005 %	
3	15	15.0 %	2.0 %	0.0002 %	
4	50	10.0 %	5.0 %	0.0001 %	
$d_a = 5$	20	7.5 %	30.0 %	0.00001 %	WAN backbone cache
$d_a<j<d_{s-1}$	-	0	0	0.00001 %	No caches within backbone
$d_{s-1} = 13$	-	80.0 %	80.0 %	0.0002 %	Cache accelerator
$d_s = 14$	-	100.0 %	100.0 %	0.001 %	Origin server

Table 2 is showing some initial results of applying the model for a caching hierarchy as introduced above. Six different configurations with respect to the number of caches available and the amount of cache refresh (IMS) requests are listed.

Table 2. Applying the model to different recipient-side configurations

	5 Caches		2 Caches		1 Cache	
IMS	10%	20%	10%	20%	10%	20%
hops/URL	6,75081	7,64484	9,6357	10,49201	9,94194	10,81204
bytes/URL	54006,460	61158,749	77085,59	83936,078	79535,529	86496,326
failures/URL	0,00265	0,00294	0,00420	0,00442	0,00469	0,00486

After developing our model to be used in a pure cache setting, it is now extended to a replication-enhanced Web. In this context, we refer to data as being replicated if it is always found in a strictly up-to-date version within hosts other than its origin server. We do not care about how this property is implemented though some proposals have been made (cf. section 2). The only important constraint in our model is that replicated data are held within any host marked with a dot in fig. 8.

Changes

In order to derive the equations for the differentiating values for average overall network hop count, bandwidth, and failure rate, some new parameters specific to our replication model have to be introduced. The first and most important one is the overall number of documents in the Web, Doc_{Num}. P_{Change} denotes the probability for a modification to one particular document during any single day. Document popularity is measured in that the top P_{Top} percent of all documents generate the $P_{Overall}$ percentage of overall WWW traffic. In our model, $P_{Overall}$ denotes the percentage of all documents that are replicated from their origin server to other proxies between the levels 1 and d_a. The actual area of distribution of any particular document is determined by the parameters $\text{Replication}_{\text{Level}}$ and $\text{Replication}_{\text{Percent}}$. The first one denotes the actual level down to which the P_{Top} most interesting pages have been replicated. $\text{Replication}_{\text{Percent}}$ in turn determines the percentage of documents replicated from level i to level i-1. The latter parameter permits to model conservative replication strategies with more limited resources closer to the clients, i.e., level 0.

The overall reduction in average network hops can now be determined by the number of hops that are not taken due to the availability of replicated documents at lower levels j. Accordingly, the reduction with respect to network hops is determined by

$$H_{red} = P_{Overall} \cdot \left((H_{d_s} + H_{d_{s-1}}) + \sum_{j = \text{Replication}_{\text{Level}}}^{d_a} (\text{Replication}_{\text{Percent}}{}^{j} \cdot H_j) \right)$$

Liberally spoken, the term within the sum denotes the number of network hops saved due to the availability of replicated data at lower levels j, i.e., $\text{Replication}_{\text{Level}}$. As already stated above, in this model the same percentage of documents is replicated from level to level down closer to the client. That is why the amount of replicated documents at any particular level beneath d_a is accordingly reduced by the factor $\text{Replication}_{\text{Percent}}$, yielding the exponential expression in the equation above.

The improvement in terms of failure reduction is comparably determined as

$$_{Red} = P_{Overall} \cdot \left((S_{d_{s-1}} + S_{d_s}) \cdot \left(\sum_{j = d_{a-1}}^{d_{s-2}} F_j + F_{d_{s-1}} \cdot P_{Acc} \right) \right) + S_{d_s} \cdot F_{d_s} + \sum_{j = \text{Replication}_{\text{Level}}}^{d_a} \text{Replication}_{\text{Percent}}{}^{j} \cdot E_j \quad .$$

The amount of bandwidth sacrificed for this degree of replication closer to the clients is determined by

$$B_{Inc} = P_{Change} \cdot Doc_{Num} \cdot Doc_{Size} \cdot P_{Top} \cdot \left(d_s - d_a + \prod_{j = \text{Replication}_{\text{Level}}}^{d_a} (\text{Replication}_{\text{Percent}}{}^{j} \cdot \text{Arity}_j \cdot (1 + j \cdot dist_r)) \right)$$

Again, the product represents the number of links traversed for the distribution of the replicated documents down to $\text{Replication}_{\text{Level}}$ multiplied with the actual amount of documents treated this way.

6. Applying the model to replication

With the set of equations derived in the previous section, it is now possible to determine the amount of overhead, resp. the exact numbers of improvement due to data replication. This section aims at giving an overview on the influence the different parameters exert on the overall numbers for bandwidth requirements, network hops per request, and error probability. As an initial example for the validity of the model, the most extreme case of replication is considered. Assuming that every single document of the Web (P_{Top}=$P_{Overall}$=100%) is replicated over all levels to the client ($\text{Replication}_{\text{Level}}$=1, $\text{Replication}_{\text{Percent}}$=100%), the improvements E_{Red} and H_{Red} —necessarily— both equal 100%.

For the more general case, all figures shown below are based on the same fixed set of parameters as introduced in table 1. In addition, the further —replication-specific— parameters listed in table 3 are used. Finally, table 4 shows the document popularity as determined in [BB+97] encoded in the parameters P_{Top} and $P_{Overall}$ used on the x axis of the right hand side of the following figures.

Table 3. Parameters used for evaluating the replication influence

P_{IMS}	$dist_r$	P_{Change}	P_{Top}	$P_{Overall}$	Doc_{Num}
10%	1	5%	1%	10%	200 mio.

Table 4. Popularity of documents

$\mathbf{P_{Top}}$	0.25 %	0.5%	1.0%	2.0%	3.0%
$\mathbf{P_{Overall}}$	6.0%	8.0%	12.0%	17.0%	22.0%

In fig. 9, the left hand side shows the percentage reduction in average network hops per request as documents are replicated to a higher degree ($\text{Replication}_{\text{Percent}}$ per level) farther to the client ($\text{Replication}_{\text{Level}}$). The right hand side underlines the advantage of replicating a higher amount of the most popular documents for different choices of the parameters $\text{Replication}_{\text{Level}}$ and $\text{Replication}_{\text{Percent}}$.

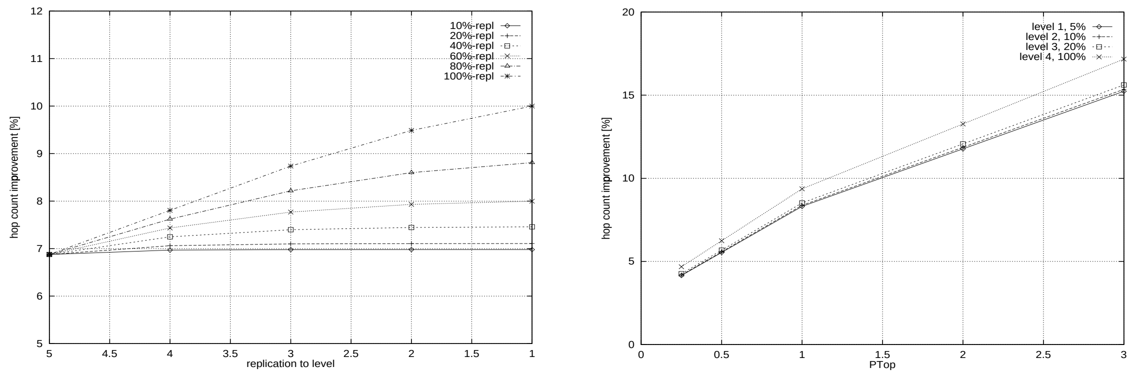


Figure 9. Overall access latency as measured in average network hops per request.

As far as the reduction of failures to retrieve documents are concerned, fig. 10 shows an even more pronounced advantage of replication. In particular, the left hand side underlines the advantages of replicating data as close to the client as possible. The improvements depicted on the right of fig. 10 again suggest that different choices of $\text{Replication}_{\text{Percent}}$ and $\text{Replication}_{\text{Level}}$ are not as important as the determination of the most valuable, i.e., most popular documents.

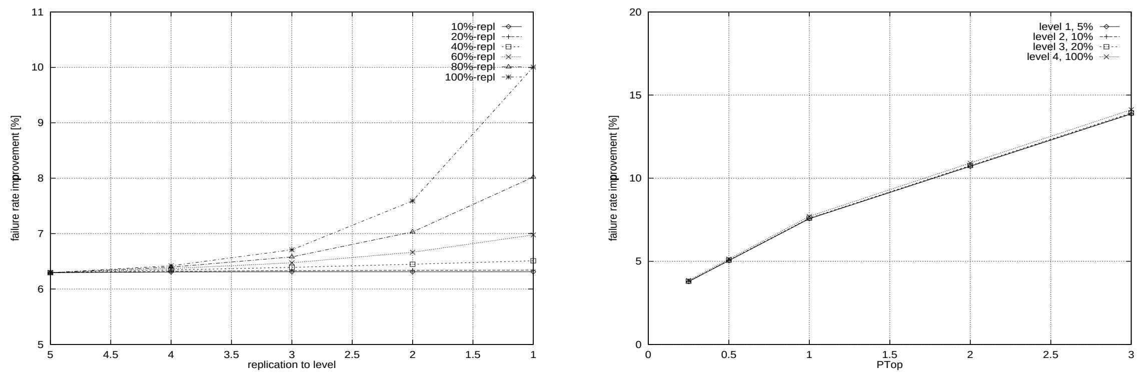


Figure 10.Overall failure rate reduction.

Finally, fig. 11 shows the major drawback of replication in the setting underlying our model where the data is replicated via broadcast closer to the clients. In each case, the bandwidth requirements are directly proportional to the amount of documents replicated, their change frequency and the degree of the spanning tree beneath d_a. This observation is certainly not surprising, but the degree to which the percentage of overhead can be controlled is somewhat interesting.

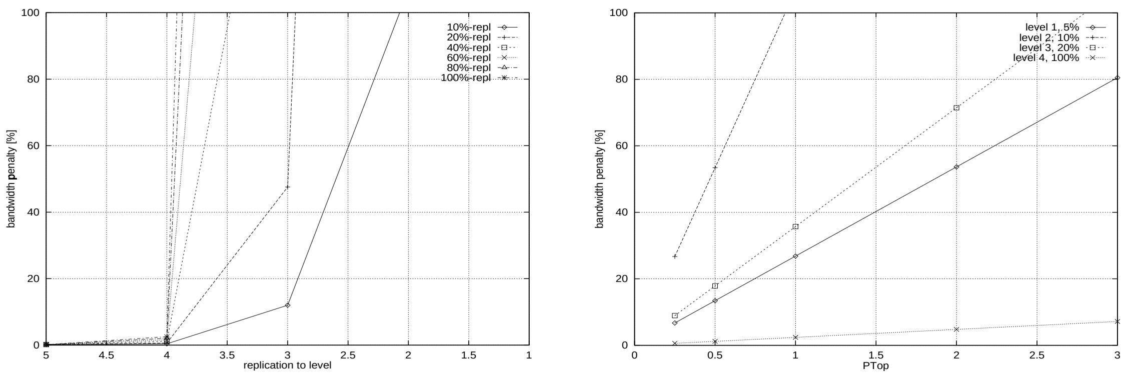


Figure 11.Overall bandwidth consumption due to replication.

In general, the biggest improvements in terms of latency/hop count reduction and failure tolerance can be gained by replicating the most popular documents to the edges of the Internet backbone structure. Only under certain circumstances replication further down the hierarchy of network links —closer to the client— is justifiable. With regard to the amount of bandwidth required for such a feat, some caveats have to be issued. To begin with, the document change rate of the most profitably replicated documents determines the amount of network traffic induced to the largest degree. Together with the specific recipient side configuration with respect to the arity on the single levels, this can prohibit active replication approaches from being successful. Nevertheless, our model presented in this paper can be used to determine which specific configurations are suitable for data replication. Finally, if it were possible to remove the main penalty of replication, i.e., the induced bandwidth consumption from the network itself, advantages quantifiable by this model can be reaped. One approach would be the regular dissemination of the most popular data via a satellite system in much the same way Usenet News are being distributed to some recipients.

7. Further Work

As far as the trace-based simulations are concerned, three immediate directions for further work are seen. The first one concerns experimentation with different caching strategies, like the one employed by the most recent Squid cache version [Wes97]. Another parameter to be looked more closely into is the failure rate as visible to a single proxy. The model introduced in this paper suggested a surprisingly high ability of caching and replication in the Web to mask failures from the user. By enhancing the simulator with the capability to record failures within single document retrievals, we are confident to find out about the actual numbers of errors masked. Finally, a more challenging task is the extension of our simulator to complete caching hierarchies. The main obstacle for this, however, is the non-availability of logfiles for most proxies of a caching hierarchy during identical intervals, though there are notable exceptions [NLANR].

In general, the results reported in this work have to be considered with the explicit simplifications made as stated in sections 4 and 5. Though they were necessary to make simulation and modeling viable, some of them might have to be refined. In addition, it seems to be feasible to also model the potential for server load reduction by means of data replication. The reduction in terms of reduced server hit count and bytes transferred can be easily deduced from the model suggested in this work. Nevertheless, the actual means of replication and the still rather simple modeling of the recipient side caused us to refrain from doing this at this early stage of development of our model for caching and replication behavior in the Web. Finally, another question arose from our work, though it cannot be answered directly with the particular model suggested in this paper: Is it possible to develop a precise algorithm determining the number of clients’ requests necessary to make replication beneficial in all values discussed above? To name just a few, such an algorithm would have to contain explicit dependencies between document popularity, change frequencies of those pages, and actual costs for network utilization on different levels of the Internet structure. Unfortunately, these values are either very hard to determine, or cannot be specified at all in the current Internet. Nevertheless, such questions are going to be interesting to answer in a policy-routable IPng [BrM96].

8. Conclusion

This paper presented a two-tiered approach to quantifying the use of caching and replication techniques for the World-Wide Web. Both approaches are based on the evaluation of real-world traces of the logfiles of WWW caching proxies. The first approach consisted of a simulation of the effects various caching strategies, cache sizes, and replication parameters have on a single WWW proxy. Secondly, an application-level model has been presented that focuses on the overall effects of intertwined caching and replication concepts on the Web. This model —available online at [Bae97]— makes explicit the relationships between degree of replication, document popularity, cache hit rates, error rates, resulting overall network bandwidth, network hops, and overall failure rates. For example, under certain circumstances, latency and failure rates per request can be reduced by about 7% while at the same time increasing the bandwidth requirements by only 2%. In general, the most interesting overall contributions of this work are the explicit dependencies found between the resultant values for bandwidth, latency, and failure rates for a replication-enhanced WWW. In this respect, a tool set has been developed that permits to make conclusive decisions on what kind of caching and replication infrastructure to use for the WWW.

References

[AS+95] M. Abrams, C.R. Standridge, G. Abdulla, S. Williams, E.A. Fox: "Caching Proxies: Limitations and Potentials"; *Proceedings 4th International Conference on the World-Wide Web*, Dec. 1995.

[Bae97] M. Baentsch: "A hands-on model for estimating effects of caching and replication systems for the WWW"; http://www.uni-kl.de/AG-Nehmer/baentsch/model.html, 1997.

[BB+97] M. Baentsch, L. Baum, G. Molter, S. Rothkugel, P. Sturm: "World-Wide Web Caching - The Application level view of the Internet"; (accepted for publication in) *IEEE Communications*, 1997.

[Be95a] A. Bestavros: "Using Speculation to Reduce Server Load and Service Time on the WWW"; Technical Report TR-95-003, Boston University, Comp. Sci. Department, Feb. 1995.

[Be95b] A. Bestavros: "Demand-based Resource Allocation to Reduce Traffic and Balance Load in Distributed Information Systems"; *Proceedings 7th IEEE Symposium on Parallel and Distributed Processing*, Oct. 1995.

[BC+95] A. Bestavros, R.L. Carter, M.E. Crovella, C.R. Cunha, A. Heddaya, S.A. Mirdad: "Application-Level Document Caching in the Internet"; *Proceedings 2nd International Workshop on Services in Distributed and Networked Environments*, Jun. 1995.

[BMS96] M. Baentsch, G. Molter, P. Sturm: "Introducing Application-level Replication and Naming into today's Web"; *Computer Networks and ISDN Systems, Vol. 28, No. 6; Proceedings 5th International Conference on the World-Wide Web*, May 1996.

[Bol93] J.-C. Bolot: "End-to-End Packet Delay and Loss Behavior in the Internet"; *Proceedings of SIGCOMM'93*, Sep. 1993.

[BrM96] S.O. Bradner, A. Mankin (eds.): "IPng — Internet Protocol Next Generation"; *Addison-Wesley*, 1996.

[CD+96] A. Chankhuntod, P.B. Danzig, C. Neerdaels, M.F. Schwartz, K.J. Worrell: "A Hierarchical Internet Object Cache"; *Proceedings of USENIX 1996 Winter Conference*, Jan. 1996.

[Chi93] B. Chinoy: "Dynamics of Internet Routing Information"; *Proceedings of SIGCOMM'93*, Sep. 1993.

[CPB93] K.C. Claffy, G.C. Polyzos, H.-W. Braun: "Traffic Characteristics of the T1 NSFNET Backbone"; *Proceedings of SIGCOMM'93*, Sep. 1993.

[DHS93] P.B. Danzig, R.S. Hall, M.F. Schwartz: "A Case for Caching File Objects Inside Internetworks"; *Proceedings of SIGCOMM'93*, Sep. 1993.

[Don95] J.E. Donnelley: "WWW media distribution via Hopwise Reliable Multicast"; *Computer Networks and ISDN Systems, Vol. 27, No.5; Proceedings 3rd International Conference on the World-Wide Web*, Apr. 1995.

[GwS95] J.S. Gwertzman, M. Seltzer: "The Case for Geographical Push Caching"; *Proceedings 5th Workshop on Hot Topics in Operating Systems*, May 1995.

[GwS96] J.S. Gwertzman, M. Seltzer: "World-Wide Web Cache Consistency"; *Proceedings of USENIX 1996 Winter Conference*, Jan. 1996.

[HK+88] J.H. Howard, M.L. Kazar, S.G. Menees, D.A. Nichols, M. Satyanarayanan, R.N. Sidebotham, M.J. West: "Scale and Performance in a Distributed File System"; *ACM Transactions on Computer Systems*, Vol.6, No.1, Feb. 1988.

[Lau96] A. Lauer: "Trace-based simulation of WWW access behavior for determining optimal replication strategies"; Master's thesis (german) and sourcecode; http://www.uni-kl.de/AG-Nehmer/GeneSys/lauer/thesis.html, Oct. 1996.

[LuA94] A. Luotonen, K. Altis: "WWW Proxies"; *Computer Networks and ISDN Systems, Vol. 27, No. 2; Proc. 1st International Conference on the World-Wide Web*, May 1994.

[MLB95] R. Malpani, J. Lorch, D. Berger: "Making World Wide Web Caching Servers Cooperate"; *Proceedings 4th International Conference on the World-Wide Web*, Dec. 1995.

[NLANR] Online Logfiles of several caching proxies in the NLANR Distributed Testbed for National Information Provisioning: ftp://oceana.nlanr.net/Traces/Caching/.

[Pax96] V. Paxon: "End-to-End Routing Behavior in the Internet"; *Proceedings of SIGCOMM'96*, Aug. 1996.

[PiR94] J.E. Pitkow, M.M. Recker: "A Simple Yet Robust Caching Algorithm Based on Dynamic Access Patterns"; *Proceedings 2nd International Conference on the World-Wide Web,* Oct. 1994.

[VNo92] H.J. Van Norman: "The case for simulation and modeling in the internetworking environment"; *Telecommunications*, Vol.26, No.9, Sep. 1992.

[WaC96] Z. Wang, J. Crowcroft: "Prefetching in World Wide Web"; *Proceeedings IEEE Global Internet 96*, Nov. 1996.

[WA+96] S. Williams, M. Abrams, C.R. Standridge, G. Abdulla, E.A. Fox: "Removal Policies in Network Caches for World-Wide Web Douments"; *Proceedings of SIGCOMM'96*, Aug. 1996.

[Wes95] D. Wessels: "Intelligent Caching for World-Wide Web Objects"; *Proceedings INET'95*.

[Wes97] D. Wessels: "Squid Internet Object Cache"; Online documentation and software at http://squid.nlanr.net/, 1997.

[ZCB96] E.W. Zegura, K.L. Calvert, S. Bhattacharjee: "How to model an internetwork"; *Proceedings IEEE INFOCOM'96*, Mar. 1996.