Efficiency Considerations for Multicast Web Caching

Josef Schmidbauer and Hilmar Linder
Department of Scientific Computing
Paris-Lodron University of Salzburg
Salzburg, Austria
Email: [jschmid,hlinder]@cosy.sbg.ac.at

***Abstract*— Wireless broadband networks are an attractive way of connecting users to the Internet. The services offered by such networks typically are point-to-point services that do not utilize the broadcast capabilities of the wireless networks. Multicast Web Caching is a readily-available and deployed method that embraces the broadcast capabilities of wireless links and largely improves web transfers.**

In this paper, we analyze the efficiency of our Multicast Web Caching solution. The suitability is proven by trace-based simulation that focus on the reductions in connection latency as well as bandwith savings on both forward an return link.

I. Introduction

Wireless networks often use forward links with broadcast capabilities to connect a central access point with a number of client nodes. Due to this broadcast nature, each client node can receive the whole traffic on the link, and implementations of common point-to-point services such as IP need to filter out the packets destined for the local IP address. All other packets are usually dropped in the receiver hard- or software. However, some of these dropped packets may contain information which might be useful for other nodes too; i.e. some point-to-point services, such as HTTP [1], can be easily extended to support point-to-multipoint operation.

Multicast Web Caching (MC) is a technique to store a selection of all received HTTP content in a local cache, whether requested by the local node or by other nodes. As a result, frequently requested content is significantly more often available locally and does not need to be fetched from the Internet, which results in a notably lower latency during browsing the web. Moreover, less bandwidth on the broadcast link is required because of this local availability, reducing the number of required transmissions. Therefore, providers can support more users with the same link capacity [2]–[6].

HTTP does not provide direct support for broadcast links. Multicast Web Caching is therefore usually implemented by a distributed set of proxy cache applications that act as HTTP intermediaries in the request chain. Here, the server proxy cache at the central access point communicates with all the client proxy caches at the client nodes over the broadcast network utilizing IP multicast. The IMPPS software developed during the EMBRACE project serves exactly this purpose [7]. It acts as a transparent proxy that terminates the HTTP/TCP connections at both the client node and the access point. Instead of using TCP over the wireless link, the Restricted Reliable Multicast Protocol (RRMP) is used [8]. IMPPS provides fast caches that allow clients to access content received via multicast at a later time. Caches can also be filled proactively with complete web-site snapshots.

In this paper we analyse the efficiency gain of Multicast Web Caching (MC) compared with systems without caching (NC) and conventional Hierarchical Caching architectures (HC).

Other work in this field differs from our own in two significant ways. First, have used trace-based simulations with inhomogenous, real-world proxy traces to obtain more praxis relevant results. Second, in opposite to the above work, we also analysed return channel traffic which is especially important for two-way satellite and BFWA networks.

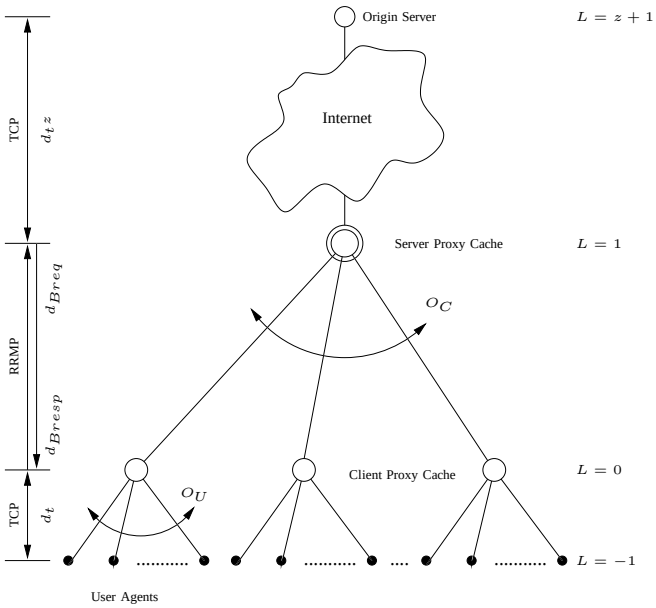

Fig. 1. Network Model.

II. Analysis

We use trace-based simulations to analyze the efficiency of the different systems. For this comparison, the connection-time and the amount of data transfered via the broadcast link from the server to a certain client are of interest. Additionally, we will compare the amount of data which must be transfered from the client to the server. This return channel traffic is important, especially in wireless networks where often an expensive channel allocation is required.

A. Network Model

We assume a strict hierarchical network architecture with a central access point as shown in figure 1. The used analytic model is very similar to the one used by Rodriguez, Spanner and Biersack for analyzing the efficiency of hierarchical and distributed caching architectures [9].

The server proxy cache running on the access point is connected with O_C different client nodes via broadcast network. Each of these client nodes is used by O_U HTTP user agents. These may be different user applications on the same hardware or various applications on different devices connected with the client proxy via a local network. The total number of users U in the system is $U = O_C \cdot O_U$.

Further we assume that the content is fetched from an origin server in a distance of z hops from the server cache. Moreover, we suppose

the distance between the user agents and the client proxy cache to be one hop. Each hop in a terrestrial network is assumed to cause a total latency of d_t. The broadcast network is assumed to have a broadcast forward link latency of d_{Bresp} and a return link latency of d_{Breq}.

For the analysis we assume that each link has infinite capacity and no cache has any storage size constraints. Thus no queueing effects or capacity-based cache misses are considered. Content expiration is the only way to remove content from the cache.

For non-caching systems, we assume that both, the access point and the client nodes does not contain any caching application.

For hierarchical caching, we assume a conventional unicast proxy cache at the access point, called server proxy cache, and one at each client nodes, called client proxy cache. The communication between the client proxy caches and the server proxy cache is based on unicast TCP connection.

For multicast operation, we further assume that a reliable multicast transport protocol was used on the broadcast link between these two caches.

We used the Restricted Reliable Multicast Protocol (RRMP) [8] for this purpose. RRMP provides nearly error free delivery of data by employing negative achknowledgements and Reed-Solomon based forward error correction techniques. Furthermore RRMP applies rate control to outgoing traffic as well as multicast congestion control.

RRMP can be also used to establish point-to-point connections. Thus non-caching and hierarchical caching was simulated using both, TCP and TCP+RRMP. The origin servers and the user agents are considered to use standard TCP.

For all caching systems we assume that each of the HTTP user agents uses a conventional local HTTP cache, e.g. the browser cache.

B. Trace-based Simulation

We used proxy log files from the IRCache Project [10] for trace-based simulations. The log files where used to calculate the popularity distribution of the different requests for each of the IRCache proxy servers. The used proxy logs cover 11.941.475 user requests for 4.258.072 different documents during 20. and 21. April 2004 for all 10 IRCache proxies. There are about 100 disjunct IP addresses in the logs of each proxy cache. We assumed that each of these addresses is linked with a unique user. The requests logs from the 10 different IRCache proxies were used to simulate 10 different client proxy caches for the network model used. The simulation was performed for one of these 10 client proxy caches which is known as "bo1" [10].

For the simulation, we assumed that each document is cacheable and has an average expiry period Δ of 24 hours. We further assumed a rather high round-trip-time of 750 ms for the broadcast link. Such high value can be found in DVB-RCS satellite systems where rather complex channel allocation algorithms are involved. We assume a hop-to-hop delay d_t of 15 ms for all other links as well as a distance to the origin server z of 10 hops.

C. Connection Time

First we calculate the required connection time $E[T_c]$. This is the time that elapse between the request and the arrival of the first part of the requested document.

1) No Caching: For a system without caching, the estimated connection time is independent of the popularity of a request. It is the sum of the connection times of all entities in the system.

Thus the connection time $E\left[T_c^{NT}\right]$ for a system without caching is given by

$$E\left[T_c^{NT}\right] = 4d_t + 2d_{Breq} + 2d_{Bresp} + 4d_t z$$

where $4d_T$ and $2d_{Breq} + 2d_{Bresp}$ is due to the three-way handshake of the TCP connection [11] that increases the number of links traversed before any data packet is sent. RRMP is negative acknowledgement based and uses no three-way handshake in order to avoid additional latencies. Therefore the connection time for the RRMP enabled link is $d_{Breq} + d_{Bres}$ (assuming no packet loss and no congestion). The resulting connection time is given by

$$E\left[T_c^{NR}\right] = 4d_t + d_{Breq} + d_{Bresp} + 4d_t z.$$

The only difference between the above two expressions is the factor 2 for the broadcast link. To shorten the following calculations, d_B will be used as the time for connection establishment over the broadcast link: $d_B = 2\left(d_{Breq} + d_{Bresp}\right)$ if TCP is used on the broadcast link or $d_B = d_{Breq} + d_{Bresp}$ if RRMP is used.

2) Hierarchical Caching: For hierarchical caching, the calculation is more complex as the object might be cached by a intermediary and doesn't need to be fetched from the origin server in this case. For this calculation we used a model which is analog to the model used by Rodriguez and Biersack [9].

Let L_i be the number of links a request for document i travels before it is satisfied by the client proxy cache, the server proxy cache or the origin server.

The connection time $E[T_c]$ in a hierarchical system is given by

$$E[T_c] = 4d_t P(L_i \geq 0) + d_B P(L_i \geq 1) + 4d_t z P(L_i \geq 2). \quad (1)$$

We now calculate the distribution of L_i. $P(L_i \geq l)$ is the probability that the number of links traversed to retreive the document is equal to l or higher. To calculate $P(L_i \geq l)$, let τ denote the time into the interval $[0, \Delta]$ at which a request occurs where Δ is average expiry time of a document. The random variable τ is uniformly distributed over the interval, thus we have

$$P(L_i \geq l) = \frac{1}{\Delta}\int_0^{\Delta} P(L_i \geq l \mid \tau)\, d\tau \quad (2)$$

where $P(L_i \geq l \mid \tau)$ is the probability that there is no request for the document i in the subtree below level l during the interval $[0, \tau]$.

The probability of k hits within a interval τ for a Poisson distribution is given by

$$P(X_t = k) = \frac{(\lambda\tau)^k}{k!} e^{-\lambda\tau}$$

where λ is the average request rate per time unit. Thus the probability of no requests within the interval τ is given by

$$P(L_i \geq l \mid \tau) = e^{-\lambda_{l-1,i}\tau} \quad (3)$$

where $\lambda_{l,i}$ is the average request rate per time unit at level l for a document i which is equal to

$$\lambda_{l,i}^H = \begin{cases} l = -1: & \lambda_{U,i} \\ l = 0: & \lambda_{C,i} = O_U \cdot \lambda_{U,i} \\ l \geq 1: & \lambda_{S,i} = O_C \cdot \lambda_{C,i} \end{cases} \quad (4)$$

and $\lambda_{U,i}$ is the average request user rate for a document i.

Thus the connection time $E\left[T_{c,i}^H\right]$ for a hierarchical caching system is given by combining (1), (2), (3) and (4):

$$E\left[T_{C,i}^{H}\right] = \begin{array}{l} 4d_t \frac{1}{\lambda_{U,i}\cdot\Delta}\left(1-e^{-\lambda_{U,i}\cdot\Delta}\right) + \\ d_B \frac{1}{O_U\cdot\lambda_{U,i}\cdot\Delta}\left(1-e^{-O_U\lambda_{U,i}\cdot\Delta}\right) + \\ 4d_t z \frac{1}{O_C\cdot O_U\cdot\lambda_{U,i}\cdot\Delta}\left(1-e^{-O_C O_U\lambda_{U,i}\cdot\Delta}\right) \end{array} \quad (5)$$

3) Multicast Web Caching: For multicast web caching we assume that every document will be cached in all client proxy caches connected via the broadcast link. Each cache miss on a single client cache will lead to a multicast transmission of the requested object via the broadcast link. Hence, other client caches will receive and store this document too. Therefore, there is no difference to a local cache-miss. This can be modeled by assuming $O_C \cdot O_U$ users per client proxy cache instead of O_U. Hence, in a multicast caching system, $\lambda_{l,i}^{M}$ is given by

$$\lambda_{l,i}^{M} = \begin{cases} l=-1: & \lambda_{U,i} \\ l=0: & \lambda_{C,i}^{M} = O_U \cdot O_C \cdot \lambda_{U,i} \\ l \geq 1: & \lambda_{S,i} = \lambda_{C,i}^{M} \end{cases} \quad (6)$$

$E[T_{C,i}^{M}]$ is calculated analog to (5).

Note that $P(L_i \geq 1) = P(L_i \geq 2)$. Therefore, each request which travels to the server proxy cache will also travel to the origin server. The resulting cache hit ratio for the server proxy cache is

$$P(L_i = 1) = P(L_i \geq 1) - P(L_i \geq 2) = 0.$$

Hence, there is no need for a server side cache in a full multicast caching system under the assumption of infinite large client caches. As real caches have disk space restrictions and limited processing capacity, the real client proxy cache hit ratio will be smaller than $P(L_i = 0)$. Thus a server proxy cache is able to improve system efficiency by caching this content.

The latency improvement compared with multicast web caching can be defined as

$$\eta_{T_{C,i}}^{H\rightarrow M} = \frac{E\left[T_{C,i}^{H}\right]}{E\left[T_{C,i}^{M}\right]}$$

If all parameters including the number of users $U = O_C \cdot O_U$ are kept constant, $\eta_{T_{C,i}}^{H\rightarrow M}$ depends most on the number of client caches O_C in the system. If there is only one client cache in the system ($O_C = 1$), then the efficiency factor $\eta = 1$. This behavior is expected, as a multicast cache system with only one multicast receiver should perform like a hierarchical caching system. If the number of client caches O_C within the multicast web caching system increase, the system efficiency η will also raise. Thus the latency improvement compared with hierarchical caching is maximal in a system with U client caches and only 1 user per client (See figure 3).

4) Results: Figure 2 shows the required connection time for fetching document i, depending on the request rate $\lambda_{U,i}$. The figure contains the results for hierarchical caching and multicast web caching. The connection time for non-caching systems is not shown as it is constant, independent of the request rate.

Figure 3 shows the connection time of hierarchical caching compared with multicast caching, depending on the number of users U.

Figure 4 shows the improved connection time for a single document i compared with multicast caching.

As expected, the efficiency of multicast caching raises with the number of client proxy caches O_C. The efficiency gain is lower if there is number of users which use the same client proxy cache (O_U) raises. The higher the request rate, the more the connection time is reduced by the local user caches. Thus the efficiency gain using multicast web caching has a maximum at medium request rate as observable in figure 4.

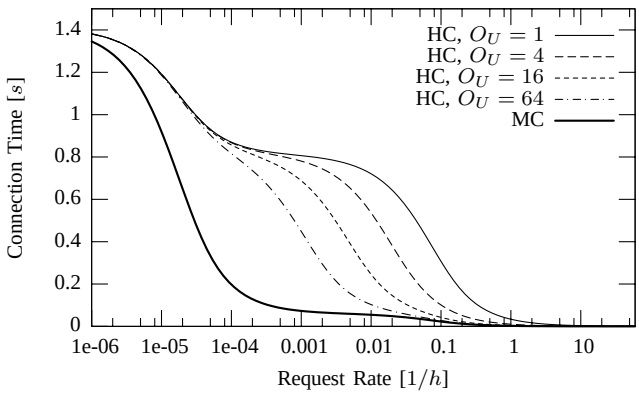

Fig. 2. Average connection time $E\left[T_{C,i}\right]$ of a hierarchical caching architecture and multicast caching, depending on the user request rate $\lambda_{U,i}$. Plotted for different number of users per client proxy cache O_U (Total number of users $U = O_C \cdot O_U = 4096$, $d_t = 15ms$, $d_B = 750ms$, $z = 10$, $\Delta = 24h$, RRMP is used on the broadcast link)

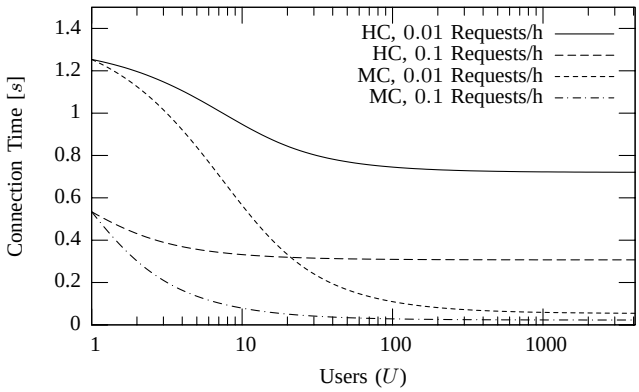

Fig. 3. Average connection time $E\left[T_{C,i}\right]$ of a hierarchical caching architecture and a multicast caching system, depending on the number of users. (One user per client proxy cache $O_U = 1$, $d_t = 15ms$, $d_B = 750ms$, $z = 10$, $\Delta = 24h$, RRMP is used on the broadcast link)

Figure 5 shows the results of the trace-based simulation for the average connection time.

Table I shows the simulated average connection time for all documents requested by the bo1 client cache. It can be seen that both, RRMP instead of TCP and hierarchical caching, have a huge impact on the connection time.

Unfortunately, multicast web caching seams to provide only a small efficiency gain if compared with hierarchical caching. One reason for this is the large number of users (90) at the bo1 cache. Thus, the client cache already benefits a lot from hierarchical caching.

We performed the simulation assuming a rather high round-trip-time on the broadcast link of 750 ms, like observable in DVB-RCS systems. For most non-satellite systems, this time will be significantly lower. Thus, the difference between the simulated systems will be smaller in this case, especially between no-caching systems and hierarchical caching.

D. Data transfered over the broadcast link

Next, we calculate the amount of data transfered via the broadcast link. We consider only the forward link and no TCP or RRMP specific

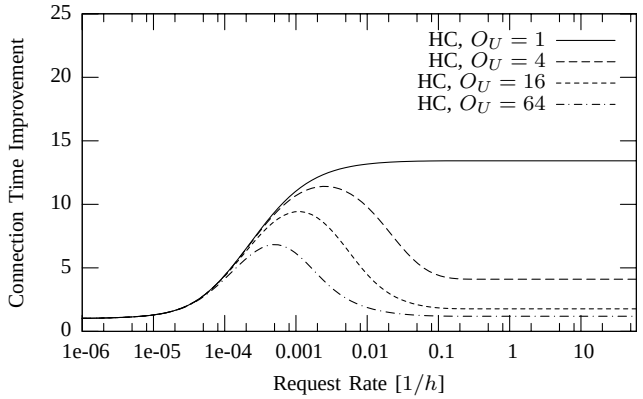

Fig. 4. Connection latency improvement $\eta_{T_{C,i}}^{H \to M}$ of multicast peer caching compared with hierarchical caching, depending on the user request rate $\lambda_{U,i}$. Plotted for different number of users per client proxy cache O_U (Total number of users $U = O_C \cdot O_U = 4096$, $d_t = 15ms$, $d_B = 750ms$, $z = 10$, $\Delta = 24h$, RRMP is used on the broadcast link)

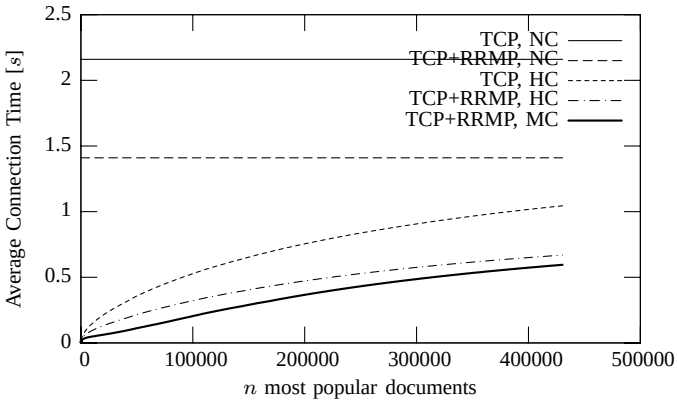

Fig. 5. Required average connection time for fetching the n most popular documents via the broadcast link. (Simulated for bo1 using the IRCache traces, $d_t = 15ms$, $d_B = 750ms$, $z = 10$, $\Delta = 24h$)

overhead for the following calculations.

The probability of a cache-miss at the client cache is $P(L_i \geq 1)$. Each client cache miss will result in a transmission of the object using the broadcast link. Hence, the required number of transmissions T_i is given by

$$T_i = O_U \cdot \lambda_{U,i} \cdot P(L_i \geq 1)$$

Further, the required bandwidth C per client proxy for all documents is given by

$$C = \sum_{i=1}^{N} S_i \cdot T_i.$$

TABLE I
AVERAGE CONNECTION TIME FOR A REQUEST FROM CLIENT CACHE 1 (BO1)

	TCP only	TCP + RRMP/MCP
w/o Caching	2.160 s	1.410 s
Hierarchical Caching	1.045 s	0.669 s
Multicast Caching	-	0.596 s

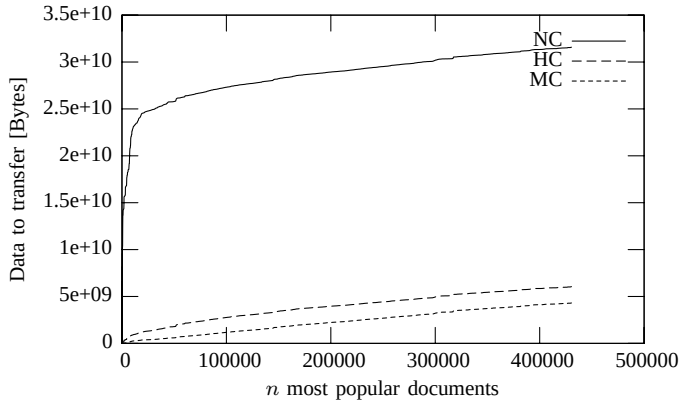

Fig. 6. Required amount of data ($\sum_{i=1}^{n} S_i \cdot T_i$) for fetching the n most popular documents via the return link (Simulated for bo1, IRCache trace based user and request distribution, $d_t = 15ms$, $d_B = 750ms$, $z = 10$, $\Delta = 24h$)

TABLE II
AMOUNT OF DATA C REQUESTED TO THE SIMULATED CACHE (BO1) VIA THE BROADCAST FORWARD LINK.

	TCP only	TCP + RRMP/MCP
w/o Caching	31.669 GByte	31.669 GByte
Hierarchical Caching	6.115 GByte	6.115 GByte
Multicast Caching	-	4.381 GByte

where S_i is the size of the i-th popular document.

But C contains only transfers triggered by the client proxy cache under observation (bo1). As already explained, in a multicast web caching system, every client node must be also able to receive the traffic generated by all other nodes. Thus, in a system of O_C client nodes where each generates an average request rate of β_C, each node must be able to receive and process responses with a rate of $O_C \cdot \beta_C$. Hence, the amount of data to receive scales linear with the number of client ($O(n)$).

1) Results: Figure 6 shows the amount of data transfered. It can be observed that non-caching systems requires more data to be transfered for a few very popular documents as expected. Hierarchical caching allows to reuse these popular documents very efficiently. Thus it needs far less bandwidth for these popular documents. Unpopular documents are used very rarely, thus the graphs having almost the same gradient for these ones. In contrast to hierarchical caching, multicast web caching can benefit from the other caches too. Hence, a lot of documents are already in the caches and need not to be transfered again. Therefore, the resulting bandwidth consumption is far lower than for hierarchical caching. Multicast web caching benefits also from less popular objects if they are used by other client proxy caches too. This explains the lower gradient of the curve.

Table II shows the amount of data which needs to be transfered via the broadcast forward link in order to respond to all requests at the bo1 client proxy cache. It can be seen that hierarchical caching can reduce the amount of data to about $\frac{1}{5}$ of the original data. Full multicast web caching can even further reduce the amount of data to less than $\frac{1}{8}$ of the original.

Note that all simulations assumed that every request is cacheable with a content expiry period of 24 hours. Non-cacheable data will require additional bandwidth which is constant for all analyzed systems. Further, we assumed that the caches have no storage limitations. Nevertheless, due to the limited user request rates and the content expiry, the used cache size has a certain maximum value [6].

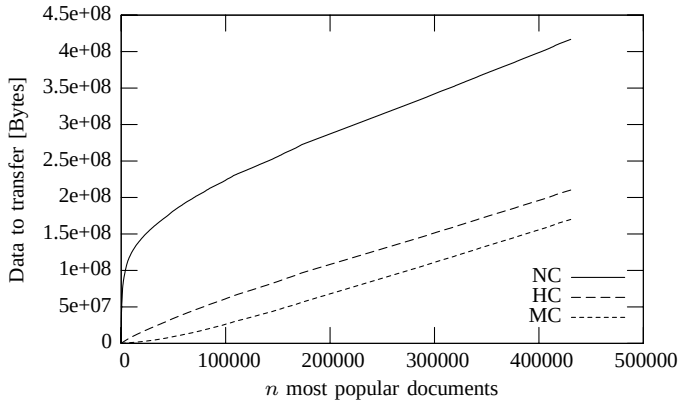

Fig. 7. Required amount of data ($\sum_{i=1}^{n} S_{R,i} \cdot T_i$) for fetching the n most popular documents via the return link (Simulated for bo1, IRCache trace based user and request distribution, $d_t = 15ms$, $d_B = 750ms$, $z = 10$, $\Delta = 24h$)

TABLE III
REQUEST DATA C_R TRANSFERED VIA THE RETURN LINK BY THE SIMULATED CACHE (BO1).

	TCP only	TCP + RRMP/MCP
w/o Caching	417.0 MByte	417.0 MByte
Hierarchical Caching	210.4 MByte	210.4 MBYte
Multicast Caching	-	170.0 MByte

E. Data transfered over the return link

In addition to the broadcast forward link, we calculated the amount of data which must be transfered via the return link. We consider only HTTP data for the calculation; no TCP or RRMP overhead.

The required bandwidth C_R per client proxy for all requested documents is given by

$$C_R = \sum_{i=1}^{N} S_{R,i} \cdot T_i.$$

where $S_{R,i}$ is the size of request header for the i-th popular document. As the used proxy traces contains no request header information, we assumed 300 Byte for the HTTP header including the most common HTTP request fields like User-Agent, Accept-Content and Accept-Encoding. Additionally we assumed that the URL string, the Referer string and the Host name string needs about 2.5 times the size of the URL string. Of course, T_i for the return link is the same as for the broadcast forward link.

1) Results: Figure 7 illustrates the return traffic for the simulated systems. The graphs are very similar to the forward link. The fact that often requested documents have a higher average size than less popular ones whereas the average request size is independent from the document rank, explains the different gradients in figure 7 compared to figure 6.

Table III shows the amount of data which need to be transfered via the return link. It can be seen that hierarchical caching can reduce the amount of data to about $\frac{1}{2}$ of the original data. Multicast web caching is able to reduce the traffic on the return channel to about 40 percent of the original data.

III. CONCLUSION

We have analysed the reduction in connection time and the bandwith savings for multicast web caching and compared it with non-caching systems and conventional hierarchical caching architectures. It could be shown that multicast web caching provides a sustainable reduction of the connection latency as well as a large reduction of the transfered data on both, the forward and the return link. The efficiency of multicast web caching heavily depends on the locality in the access patterns across the client proxies. As this locality is both, spatial and temporal and varies for each client proxy, analytical results are hard to obtain. We used trace-based simulations with real-world proxy cache log files to get realistic results. The proxy traces origin from from caches distributed all over the United States and thus represent a highly heterogeneous user community. For a more homogenous user community, for example school networks [12], the expected reductions are even larger.

ACKNOWLEDGEMENT

The authors would like to thank Duane Wessels and the IRCache Project for providing proxy cache log files. The IRCache Project was supported by the National Science Foundation (grants NCR-9616602 and NCR-9521745) and the National Laboratory for Applied Network Research. This work was partly funded by the European Commission 6th Framework Programme IST Project BROADWAN[1] and the 5th Framework Programme IST Project EMBRACE.

REFERENCES

[1] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee, "RFC2616: Hypertext Transfer Protocol – HTTP/1.1," June 1999. [Online]. Available: ftp://ftp.internic.net/rfc/rfc2616.txt,ftp://ftp.cosy.sbg.ac.at/pub/mirror/rfc/rfc2616.txt

[2] H. Linder, H. Clausen, and B. Collini-Nocker, "Satellite internet services using dvb/mpeg-2 and multicast web caching," *IEEE Communications Magazine*, vol. 38, no. 6, pp. 156–161, June 2000.

[3] H. Linder, H. D. Clausen, and J. Schmidbauer, "Secure and reliable multi-level multicast web caching for satellite networks," *ASSI Indonesian Satellite Association, Satellite Communications Letter*, vol. II, no. 2, August 2002.

[4] M. Agrawal, A. Manjhi, N. Bansal, and S. Seshan, "Improving web performance in broadcast-unicast networks," in *Proceedings of IEEE INFOCOM 2003*, 2003. [Online]. Available: http://www.ieee-infocom.org/2003/papers/06_03.pdf

[5] A. Armon and H. Levy, "Cache satellite distribution systems: Modeling and analysis," in *Proceedings of IEEE INFOCOM 2003*, 2003. [Online]. Available: http://www.ieee-infocom.org/2003/papers/06_04.pdf

[6] P. Rodriguez and E. Biersack, "Bringing the web to the network edge: Large caches and satellite distribution," *Mobile Networks and Applications*, vol. 1, no. 7, pp. 67–78, January 2002.

[7] H. Linder, H. D. Clausen, and W. Stering, "A multi-level, multicast web caching system for interconnected lmds networks," in *IST Mobile and Wireless Telecommunication Summit, Thessaloniki, Greece*, 2002.

[8] H. Linder and H. Clausen, "Scalable multicast data distribution for different transport service classes," in *Proceedings of the IEEE International Performance, Computing and Communications Conference (IPCCC 1998)*, February 1998, pp. 435–441.

[9] P. Rodriguez, C. Spanner, and E. Biersack, "Analysis of web caching architectures: Hierarchical and distributed caching," in *IEEE/ACM Transactions on Networking*, vol. 9 (4), August 2001, pp. 404–418. [Online]. Available: citeseer.ist.psu.edu/article/rodriguez01analysis.html

[10] "Hierarchical caching system usage statistics." [Online]. Available: http://www.ircache.net/statistics

[11] J. Postel, "RFC 793: Transmission control protocol," Sept. 1981, see also STD0007 [13]. Status: STANDARD. [Online]. Available: ftp://ftp.internic.net/rfc/rfc793.txt,ftp://ftp.math.utah.edu/pub/rfc/rfc793.txt

[12] "Schoolcast project, European Space Agency ESA," http://telecom.esa.int/schoolcast.

[13] J. Postel, "STD 7: Transmission Control Protocol: DARPA Internet Program Protocol Specification," Sept. 1981, see also RFC0793 [11]. [Online]. Available: ftp://ftp.isi.edu/in-notes/rfc793.txt;ftp://ftp.isi.edu/in-notes/std/std7.txt;ftp://ftp.math.utah.edu/pub/rfc/rfc793.txt;ftp://ftp.math.utah.edu/pub/rfc/std/std7.txt

[1] `http://www.broadwan.org`