Copyright 1997 IEEE. Published in the 1997 International Conference on Image Processing (ICIP'97), scheduled for October 26-29, 1997 in Santa Barbara, CA. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works, must be obtained from the IEEE. Contact: Manager, Copyrights and Permissions / IEEE Service Center / 445 Hoes Lane / P.O. Box 1331 / Piscataway, NJ 08855-1331, USA. Telephone: + Intl. 908-562-3966.

Soft Caching: Image Caching in a Rate-Distortion Framework

Claudio Weidmann and Martin Vetterli

Dept. of Electrical Engineering
Swiss Federal Institute of Technology,
Lausanne, Switzerland
{weidmann, vetterli}@lcav.de.epfl.ch

Antonio Ortega and Fabio Carignano

Integrated Media Systems Center,
University of Southern California,
Los Angeles, CA
ortega@sipi.usc.edu

Abstract

This paper presents a novel approach to image caching for image databases, web browsers, proxies and other similar applications. Current caches employ a hard strategy: either the image is stored in the cache, or it is not. In a soft cache, a variable amount of memory is assigned to each image. This is ideally matched to progressive image file formats. Our strategy for optimal soft caching considers the image download delay as a distortion measure. Then the minimization of the expected delay can be carried out in an operational rate-distortion framework. We present optimal theoretical solutions as well as simulation results.

1. Introduction

The ever-increasing traffic on the Internet is mainly generated by web browsing applications. Proxy caching, which allows some of the most popular web objects to be cached at intermediate network nodes, has been shown to provide substantial performance gains. Since images account for the largest share of web traffic both in terms of number and size, we argue that image-specific caching techniques will yield better performance than methods that treat all objects in the same manner [3].

Our analysis is based on the static model shown in figure 1, which reflects the situation for a distributed image database. In section 5 we show how this can be adapted to the highly dynamic web environment.

We consider a static set of N images of size R_1, ..., R_N bits, which are stored using a state of the art embedded code [6,7] or the standard progressive JPEG format [9]. This means that the first r_i bits of image i will yield some intermediate resolution when decoded and displayed. To recover the full resolution, just the remaining R_i - r_i bits have to be sent to the client decoder. The images could be stored on different servers; therefore to each we assign a link bandwidth B_{Si} [bit/s]. This is the speed at which image i can be retrieved from its server. The client is connected to the cache through a link of bandwidth B_C. If there are several clients, we combine them into a single client with an aggregate access behavior. This is reasonable as long as the cache serves a group of users with relatively homogeneous link characteristics (e.g. a campus network). In the following, we assume $B_C > B_{Si}$ for all images i, since otherwise caching cannot reduce the download delay. On the other hand, if the optimization goal were to minimize the expected number of bits fetched from remote servers, one would not need these values altogether. We consider just transmission delays; i.e. we assume zero latency due to request processing and cache miss overhead. In the latter case, the missing part of an image is transmitted directly to the client at B_{Si} bits/sec.

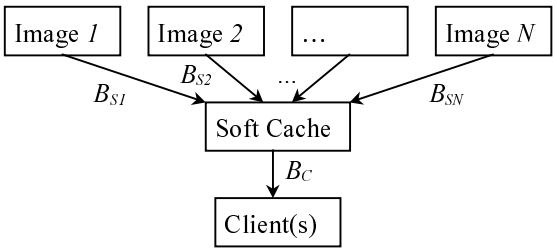

Figure 1: Static image database model

The main difference between soft and hard caching is that the former can store images in the cache at any rate between 0 (not in cache) and R_i (full resolution in cache). Of course actual implementations will be restricted to a finite set of rates. We distinguish two basic ways of accessing an image:

Hard access: The client always requests full rate R_i. If the cache holds the first r_{Ci} bits of the image, the total delay for getting image i will be

$$\delta_i(r_{Ci}) = B_C^{-1} \cdot r_{Ci} + B_{Si}^{-1} \cdot (R_i - r_{Ci}) \qquad (1)$$

Soft access: The client requests rate r_i, $0 < r_i \le R_i$. The delay is

$$\delta_i(r_i, r_{Ci}) = \begin{cases} B_C^{-1} \cdot r_i, & r_i \le r_{Ci} \\ B_C^{-1} \cdot r_{Ci} + B_{Si}^{-1} \cdot (r_i - r_{Ci}), & r_i > r_{Ci} \end{cases} \quad (2)$$

In practical soft access, the user just stops downloading data when an acceptable quality level has been reached, thus individually trading off perceptual distortion for download time (i.e. rate). This soft access behavior is what makes soft caching really pay off.

The goal of optimal soft cache allocation shall be to minimize the expected image download delay given the image access probabilities P_i, and given a cache size constraint R_C. This problem is similar to optimized multiresolution image retrieval, where the average query duration is minimized by optimal rate allocations to different resolution levels [4].

The remainder of the paper develops optimal allocation procedures in the case of static access probabilities. This is wrapped up by simulation results and a proposal for a practical soft cache architecture.

2. Hard Access

When images are always requested at full rate, the following minimization problem results:

$$\begin{aligned} &\min_{\mathbf{r}_C} E[\delta] = \min_{\mathbf{r}_C} \sum_{i=1}^{N} P_i \cdot \delta_i(r_{Ci}) \\ &\text{subject to} \sum_{i=1}^{N} r_{Ci} \le R_C \\ &\text{and } 0 \le r_{Ci} \le R_i \end{aligned} \quad (3)$$

where P_i is the access probability of image i, r_{Ci} is its rate kept in the cache, R_C the maximal cache size. With delay δ_i given by (1), we get the following simplified problem:

$$\min_{\mathbf{r}_C} -\sum_{i=1}^{N} P_i \cdot \left(B_{Si}^{-1} - B_C^{-1}\right) \cdot r_{Ci} \quad (4)$$

subject to the same constraints as in (3). The optimization problem (3) or (4) can be solved with classical linear programming methods such as the simplex algorithm [2]. Alternatively, a Lagrangian optimization is carried out along the same lines as in the soft access case explained in the next section.

The expression for delay in (1) assumes that no additional memory is available at the cache, and that the image data has to be transferred sequentially (in order). Otherwise one could start fetching the missing R_i - r_{Ci} bits from the server as soon as the request is issued, thus obviously improving the cache performance. This doesn't change the basic structure of problem (3) except for the upper bounds on the r_{Ci}.

3. Soft Access

Now, we assume that each image i can be requested at a specified rate r_i, which is taken from a finite set of rates $\{r_{i,j} : j = 1...k_i\}$ with conditional selection probabilities $P(r_{i,j}|i)$. We request $0 < r_{i,j} < r_{i,j+1} \le R_i$ and $P(0|i) = 0$, $\forall i$. Denote by $d_i(r_{Ci})$ the contribution of image i to average delay:

$$\begin{aligned} d_i(r_{Ci}) &= P_i \cdot \left[\Pr\{r_i \le r_{Ci}\} \cdot \mathrm{E}[r_i | r_i \le r_{Ci}] \cdot B_C^{-1} \right. \\ &\quad + \Pr\{r_i > r_{Ci}\} \cdot r_{Ci} \cdot \left(B_C^{-1} - B_{Si}^{-1}\right) \\ &\quad \left. + \Pr\{r_i > r_{Ci}\} \cdot \mathrm{E}[r_i | r_i > r_{Ci}] \cdot B_{Si}^{-1} \right] \\ &= P_i \cdot \Big[\sum_{r_{i,j} < r_{Ci}} P(r_{i,j}|i) \cdot (r_{Ci} - r_{ij}) \cdot B_{Ai}^{-1} \\ &\quad - r_{Ci} \cdot B_{Ai}^{-1} + \mathrm{E}[r_i] \cdot B_{Si}^{-1} \Big] \end{aligned} \quad (5)$$

where for convenience we introduced $B_{Ai}^{-1} = B_{Si}^{-1} - B_C^{-1}$, which could be interpreted as the reciprocal of the "accelerating bandwidth" for image i. The following lemma will be useful in showing the optimality of the allocation algorithm.

Lemma: $d_i(r_{Ci})$ is convex-$\cup$ for any distribution $P(r_{i,j}|i)$.

Proof:
We compute the difference sequence $\{\Delta_{i,l}: l = 1..k_i\text{-}1\}$:

$$\begin{aligned} \Delta_{i,l} &= d_i(r_{i,l+1}) - d_i(r_{i,l}) \\ &= -P_i \cdot \left(1 - \sum_{j=1}^{l} P(r_{i,j}|i)\right) \cdot (r_{i,l+1} - r_{i,l}) \cdot B_{Ai}^{-1} \\ &\le 0 \end{aligned}$$

The second term of the product is monotonically decreasing in l, i.e. $\Delta_{i,l}$ is monotonically increasing and hence $d_i(r_{Ci})$ is convex-$\cup$. □

Therefore, each image is characterized by an operational rate-distortion curve $d_i(r_{Ci})$. The cache rate allocation problem can then be stated as follows:

$$\begin{aligned} &\min_{\mathbf{r}_C} D(\mathbf{r}_C) = \min_{\mathbf{r}_C} \sum_{i=1}^{N} d_i(r_{Ci}) \\ &\text{subject to} \sum_{i=1}^{N} r_{Ci} \le R_C \\ &\text{and } 0 \le r_{Ci} \le R_i \end{aligned} \quad (6)$$

This is analogous to the bit allocation problem for a set of quantizers [8], and can be solved with Lagrangian techniques. The key fact is that, for any $\lambda \geq 0$, the solution $\mathbf{r}_C$* to the unconstrained problem

$$\min_{\mathbf{r}_C} \sum_{i=1}^{N} d_i(r_{Ci}) + \lambda \cdot \sum_{i=1}^{N} r_{Ci} \tag{7}$$

is also the solution of the constrained problem (6) with the constraint $R_C^*(\lambda) = \sum r_{Ci}^*(\lambda)$. Additionally, the minimization in (7) can be carried out separately for each image *i*. Essentially, for each $\lambda \geq 0$ one gets a set of solutions $\{\mathbf{r}_C^*(\lambda)\}$ with rates $\{R_C^*(\lambda)\}$. These solution rates are monotonically non-increasing in λ (for a proof of these facts, see [8]). Therefore a simple bisection algorithm will find the λ for which the original constraint R_C is satisfied. An example of such a procedure can be found in [5]. The bisection algorithm can only find solutions on the convex hull of the global distortion-rate curve. But the fact that the $d_i(r_{Ci})$ are convex guarantees that the optimal point lies indeed on the convex hull [8].

4. Simulation results

The above results apply to a static image database with stationary access probabilities. To simulate such a situation, we extracted the relevant parameters from trace files of WWW proxy caches. This yielded models for the distribution of image sizes, server bandwidths and access probabilities. Of course no information about soft access behavior could be gathered in that manner; therefore we took the same approach as in [4] and modeled the conditional access probabilities $P(r_{i,j}|i)$ with the following parameterized function:

$$P_m(r_{i,j} \mid i) = \left(1 - \frac{r_{i,j}}{R_i}\right)^m$$

One important finding was that the cache performance depends almost only on the cache size relative to the sum of all image sizes, i.e. R_C / sum(R_i). Hence the simulations had only to be run for a range of relative cache sizes. Further details are available in [1].

Figure 2 shows the expected delay performance of soft caching, while the ratio between delays for soft and hard caching is plotted in figure 3. Soft caching consistently outperforms hard caching, but the gain drops as the relative cache size exceeds about 0.2. The intuition behind this is that at this point both types of cache hold the set of most popular images which satisfies almost all requests. The soft cache holds more images, but the additional images are rarely requested. Hence the gain drops. In fact, for a relative cache size of 1 both caches obviously have the same performance.

To get meaningful results for a soft web proxy cache, we have modified a publicly available cache so that images are recoded in progressive JPEG format on the fly. Then we record at which rate the users stop downloading. With this data we will be able to run a trace driven simulator that closely mimics an actual web proxy.

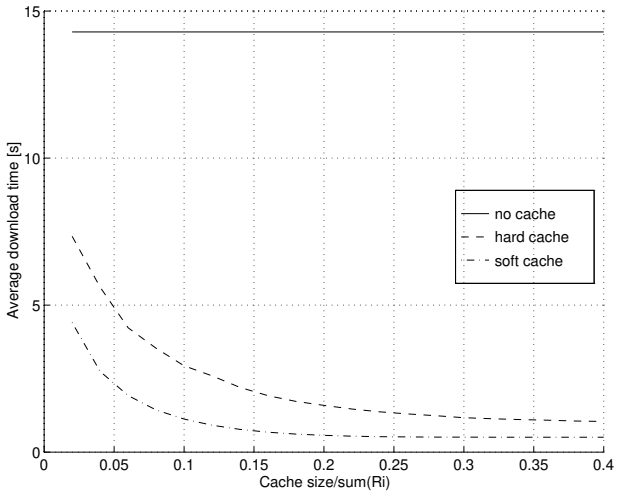

Figure 2: Average download time vs. relative cache size for a soft access soft cache (m = 4)

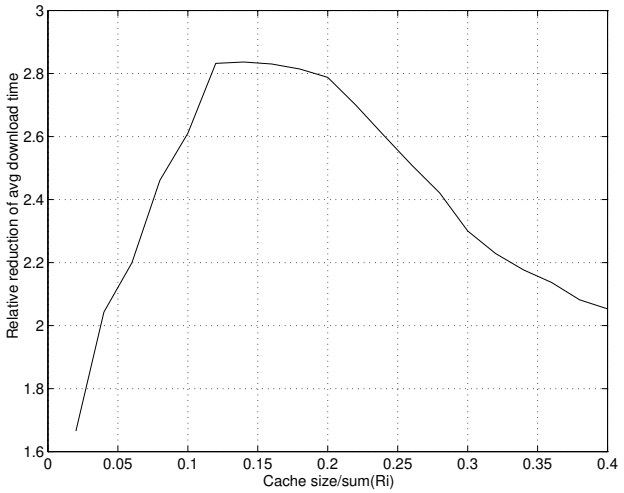

Figure 3: Relative gain of soft caching with respect to hard caching (m = 4)

5. Implementation issues

In a real system, by default every image is cached until the cache runs out of memory. At that point, a garbage collector tries to free up some memory by removing the least

costly images (in terms of increase in expected delay). This suggests a system similar to the one in figure 4, where all subsystems work concurrently.

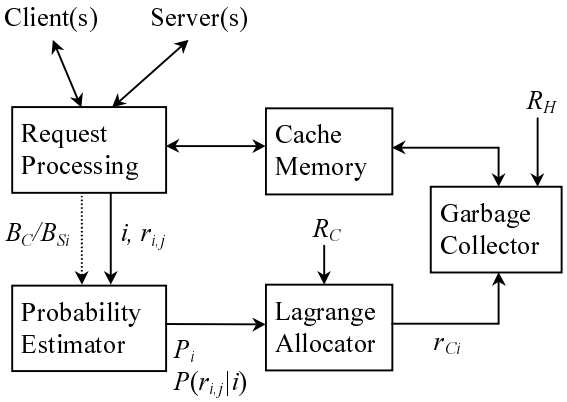

Figure 4: An adaptive caching system

On a cache miss ($r_i > r_{Ci}$), the request processor tries to store the missing image part if there is enough memory available. It then passes (i, r_{ij}) on to the probability estimator. Meanwhile, the allocator computes the optimal r_C based on a snapshot of the probability estimates. In stationary operation, the latter will change only slowly with time. This in turn implies that the optimal Lagrange multiplier λ is slowly varying too. Hence the allocation algorithm can be sped up by restricting the range of λ's to be searched. Last but not least, the garbage collector watches the cache size. As soon as a high watermark rate R_H is exceeded, it starts (partially) removing images, based on the last optimal allocation computed. The allocation target rate R_C and the watermark rate R_H are "tweak" parameters, which among others will depend on the system load.

6. Conclusion

We have introduced the concept of soft image caching and given optimal solutions in terms of classical rate allocation problems. With the increasing popularity of image formats such as progressive JPEG, soft caching seems a natural way to reduce delay and save bandwidth. The browsing will be even quicker thanks to the mere fact that a soft cache automagically delivers fast low-resolution previews. And the necessary protocol basics to fetch just a part of a file are already there, namely the range retrieval request defined in HTTP 1.1.

Our simulations show that for static access behavior significant gains with respect to hard caching are achievable. We are currently investigating the performance of a soft web cache to extend these findings to dynamic, nonstationary access patterns. Simulation and experimental results will be made available on our web sites [3] as they are generated.

Acknowledgement

We thank Gerhard Ruhl for his contributions to our web cache simulator project.

References

[1] F. Carignano, "Soft Caching: Web Cache Management Techniques for Images," *Internship Report*, USC, July 1997; available at http://biron.usc.edu/~carignan/

[2] M. Minoux, "Mathematical Programming: Theory and Algorithms," Wiley, 1986.

[3] A. Ortega, F. Carignano, S. Ayer and M. Vetterli, "Soft Caching: Web Cache Management for Images," *IEEE Signal Processing Society Workshop on Multimedia*, Princeton, NJ, June 1997; also available at http://sipi.usc.edu/~ortega/SoftCaching/index.html and http://lcavwww.epfl.ch/~weidmann/softcache/index.html

[4] A. Ortega, Z. Zhang and M. Vetterli, "A Framework for the Optimization of a Multiresolution Remote Image Retrieval System," *Infocom '94*, pp. 672--679, Toronto, June 1994.

[5] K. Ramchandran and M. Vetterli, "Best Wavelet Packet Bases in a Rate-Distortion Sense," *IEEE Trans. Image Processing*, vol. 2, no. 2, pp. 160-176, April 1993.

[6] A. Said and W. Pearlman, "An Image Multiresolution Representation for Lossless and Lossy Image Compression," *IEEE Trans. Image Processing*, vol. 5, pp. 1303-1310, Sept. 1996.

[7] J. Shapiro, "Embedded Image Coding using Zerotrees of Wavelet Coefficients," *IEEE Trans. Signal Processing*, vol. 41, pp. 3445-3462, Dec. 1993.

[8] Y. Shoham and A. Gersho, "Efficient Bit Allocation for an Arbitrary Set of Quantizers," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 36, pp. 1445-1453, Sept. 1988.

[9] G. Wallace, "The JPEG Still Picture Compression Standard," *Comm. ACM*, vol. 34, pp. 31-44, April 1991.