Locality of Reference in an Hierarchy of Web Caches

Fernando Duarte, Fabrício Benevenuto, Virgílio Almeida, Jussara Almeida

Computer Science Department
Federal University of Minas Gerais
Brazil
{fernando, fabricio, virgilio, jussara}@dcc.ufmg.br

Abstract. This work presents an extensive evaluation of the request filtering in hierarchy of proxy caches. Using the recently proposed ADF (Aggregation, Disaggregation and Filtering) model as well as entropy as metric for Web traffic characterization, we evaluate how locality of reference changes as the streams of requests pass through a hierarchy of caches. Moreover, we propose the use of average entropy for comparing the locality of reference of different streams and present how a proxy server can dynamically calculate the entropy of its incoming request stream.

Keywords: Web caching, locality of reference, entropy.

1 Introduction

The dramatic growth of network traffic and the increasing number of users are characteristics that have marked the phenomenon of the Web. The use of proxy servers has emerged as an efficient solution to increase the performance of Web systems, improving Web servers scalability and, reducing network traffic as well as the response time of user requests.

A proxy server can be seen as an intermediator of the traffic among clients and HTTP servers. When a proxy server sends a previously requested document to the clients, a copy of the document is stored in its local cache, so that future requests for this document can be directly obtained from the proxy server. These servers operate aggregating, disaggregating and filtering the request stream that passes through them. One can say they *aggregate* the arriving requests in an unique stream, which is processed using its local cache. Moreover, the proxy servers act as a *disaggregator* of traffic, distributing the arriving requests for different Web servers. When a stream of requests passes through a proxy, only the requests that could not be served from its cache are disaggregated towards the destination Web servers. In this context, one can say that a proxy acts as a request *filter*, allowing only the miss stream to be disaggregated to the rest of the Web.

Web caching is usually associated with a hierarchical organization. The browser cache, located in the user machine, is the lowest level of the hierarchy. The next

level is composed of the caches of intranets, i.e., the proxy servers in universities and organizations. Going up in the hierarchy, there are regional proxies and so forth. A request that cannot be satisfied for a proxy is immediately sent to the proxy in the next level in the hierarchy, until it reaches the destination server.

The organization of an efficient cache hierarchy involves the study of the main properties of the streams of requests. Various questions related to *caching* require a deep study of how properties of the request streams change when they pass through the proxy servers. In this context, a vision of the Web traffic according to the effects of aggregation, filtering and disaggregation associated to the appropriated metrics bring a better understanding of the effects of locality of reference in an hierarchy of Web caches.

The study of locality of reference under this new perspective of the Web was initially considered in [5]. That work considered the use of entropy as metric to measure the locality of reference of request streams. In this work, we apply adaptations of this metric in a context of cache hierarchy, evaluating the impact that different cache replacement policies have on the locality of reference of the request streams. Moreover, we propose a methodology for calculating locality of reference dynamically. The main contributions of this paper are:

1. Performance evaluation of a cache hierarchy - This work provides an extensive performance evaluation of cache replacement policies in different levels of a hierarchy of caches. We measure traditional metrics such as hit ratio and compare these results with some recently proposed metrics for locality of reference. The experiments presented allow a better comprehension of how locality of reference changes as the streams of requests pass through an hierarchy of caches. This evaluation of the locality of reference can be used as a guide for designing of hierarchy of caches.

2. Metrics to locality of reference - We propose the average entropy to allow an HTTP server or a proxy server to perceive the variation of the locality of reference of request streams. Moreover, we present how entropy can be dynamically calculated by the Web components. We believe that capturing the notion of locality in real time can be helpful for constructing self-adaptive Web caching systems.

The rest of the paper is organized as follows. The next section presents related work. Section 3 introduces entropy and the new proposed metrics. Section 4 presents the experimental methodology used in this study. Our results are detailed in section 5. Conclusions and future work are offered in section 6.

2 Related Work

Although several different definitions are currently available [1, 6, 5], it is strongly accepted that the main aspects to locality of reference are *temporal correlations* in the request streams and the *popularity distribution* of requested objects [5, 4, 7]. This work focuses on the object popularity.

The study of locality of reference was motivated by the impact of this property on the performance of cache systems. These studies were the basis for

the development of caching policies, inter-proxy communication protocols and prefetching algorithms [8].

The first attempt to characterize the impact of proxy caches on request streams is presented in [9]. Mahanti et al. studied how the temporal locality changes in different levels of a hierarchy of Web caches [6]. More recently, Williamson [10] evaluates the effectiveness of different caching policies in different levels of a hierarchy of Web caches. Whereas [10] only considered the filtering effects, [4, 5] introduced the study of two other transformations to which streams of references are submitted: aggregation and disaggregation. They grouped the three transformations into a model called ADF (Aggregation, Disaggregation and Filtering), and proposed and validated new metrics for analyzing temporal locality in a request stream moving through this model.

In this work, we evaluate the impact of locality of reference in the performance of a hierarchical caching system using the tools proposed in [5]. Moreover, we consider new forms of using the entropy proposed by [5] to analyze the locality of reference, providing a framework so that this metric can be dynamically calculated and applied to real environments.

3 Metrics for Locality of Reference

This section presents the metrics used in this paper to measure the locality of reference in streams of requests. In section 3.1 we present the concept of entropy. In section 3.2 we show an efficient form of calculating the entropy dynamically. In section 3.3, we propose the use of average entropy.

3.1 Entropy

The distribution of popularity of a set of requests usually is characterized by the *Zipf Law* [1, 3]. In general a Zipf-like distributions (the probability $P[i]$ of access the i-th most popular object is $P[i] = \frac{C}{i^{\alpha}}$, where α is a parameter and C a normalizing constant) has been used to approximate the popularity of objects in request streams in the Web. In this kind of distribution, the α coefficient is usually used as an indicator of the concentration of popularity of the request streams.

Recently, a more direct measure was proposed to evaluate the concentration of popularity of streams of requests, namely entropy [5]. The entropy $H(X)$ of a random variable X, taking n possible values with probability p_i, is calculated as follows:

$$H(X) = -\sum_{i=1}^{n} p_i \log_2 p_i \tag{1}$$

Note that $H(X)$ depends only on the probability of occurrence of the requests and the number n of different requests of the set. The maximum value ($H(X) = log_2 n$) is reached when the requests have the same probability ($p_i = 1/n, \forall i$),

and the minimum value ($H(X) = 0$) occurs when only one object concentrates all references ($p_i = 1$, $p_j = 0$ for $i \neq j$). Thus, for sets with the same number of requests, the higher the value of the entropy, the smaller the concentration of popularity in few objects.

3.2 Calculating Entropy Dynamically

We now propose a technique to dynamically compute the entropy present in a request stream. Calculating entropy dynamically allows an online analysis of locality of reference, which can be used for making operational decisions. For instance, a proxy server can vary some parameters of its configuration based on the variations that occur in the locality of reference of the arriving request stream.

In order to use the definition of entropy, proposed and validated in [5], in real environments, its value must be measured in a incremental way, being recalculated at each new arriving request. Previous works [5], have computed entropy for a set of requests, in which the number of requests was known *a priori*.

Expanding the equation 1, we find a practical and dynamic way to calculate the entropy. Let n_t be the total number of requests that have already arrived at the proxy and n_i be the number of references for the object i in that set, then p_i can be estimated as n_i/n_t. The entropy can be calculated as:

$$H(X) = \log_2 n_t - \frac{1}{n_t} \sum_{i=1}^{n} n_i log_2 n_i \tag{2}$$

Using equation 2, the entropy can be dynamically calculated keeping up to date the value of n_t and the value of the sum $S = \sum_{i=1}^{n} n_i \log_2 n_i$ for each new arriving request.

3.3 Average Entropy

The normalized entropy was proposed to compare the entropy of sets with different number of requests [5]. This normalization is based on the highest possible value for the entropy of the set of requests. Considering n as the number of distinct requests, the normalized entropy $H^n(X)$ is defined as:

$$H^n(X) = \frac{H(X)}{log_2 n} \tag{3}$$

Nevertheless, when we deal with sets of requests of equal sizes, the entropies of these sets can be compared directly, being unnecessary the normalization presented in the equation 3. Based on this observation, we propose the average entropy. We calculate the entropy of a set of requests by considering a window of m requests at a time. The window moves one request at a time, and a new entropy value is calculated. At the end, after covering the whole set of requests, we average the entropy values computed for all request windows.

Considering a window of size m, a sequence with a total of n_t requests and $H(X_{[i,j]})$ as the value of the entropy of the window that contains the interval of the i-th until the j-th request, we define the average entropy $H^m(X)$ as:

$$H^m(X) = \frac{\sum_{i=0}^{n_t-m} H(X_{[i+1,m+i]})}{n_t - m + 1} \quad (4)$$

In order to use the notion of locality of reference in a real environment such as in a Web server or in a proxy server, one needs to evaluate the locality of a certain sample of the requests that arrive at the server. In this context the average entropy, associated to the dynamically calculation of the entropy, emerge as adjusted metrics to capture the variation of popularity of the stream of requests that arrives at the servers. This can be done, for instance, by comparing the entropy of the last window with the value of the average entropy.

The size of the window for the calculation of the entropy can impact the analysis of the popularity concentration. For instance, if the window is small, the obtained entropy captures the locality of a small sample, which cannot represent correctly the popularity of the flow. On the other hand, if the size of the window is relatively high, the variation of popularity is less noticeable. We suggest that, to compare different streams of requests it is necessary that the size of the window to be of the same order of magnitude of the total of requests of the streams.

4 Experimental Methodology

This section describes the methodology used in our study. A simulator of a hierarchy of caches was built, organized as showed in figure 1 (left). This figure presents a two-level caching system, with two caches (children) on the first level and one cache (parent) on the second one. The requests made by the users are received directly by the caches in the first level, whereas the requests that cannot be satisfied in this level are aggregated forming a request stream, which is forwarded to the second level cache. There is no interaction between the first level caches.

In order to better understand the effects of the locality of reference in the hierarchy of caches, we use the ADF model [5]. This model represents the Web through a graph where the vertices are points where the request streams can be modified, and the edges are connections among these points. The vertices in the graph are of three different kinds, depending of which effect they cause in the Web traffic: Aggregation (A), Disaggregation (D) and Filtering (F).

Figure 1 (right) shows the representation of the cache hierarchy used in our experiments using the ADF model. The caches of the first level function as points of aggregation of the user requests. These caches also apply a filtering transformation on the streams of requests. The streams of missed requests that are forwarded by the first level caches are aggregated and again, are filtered in second level cache, where they are finally disaggregated to the Web servers.

In the simulations, we evaluate the behavior of the average entropy when streams of requests pass through the hierarchy of caches, varying the size of

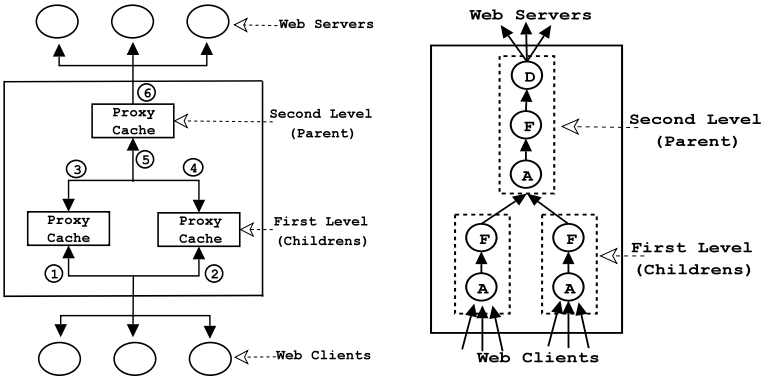

Fig. 1. System of hierarchy caches: ISP overview(*left*), ADF model(*right*)

these caches from $1MB$ to $16GB$. To choose the size of the window used in the calculation of the average entropy, several experiments were executed varying the window size. The difference of the results obtained for window sizes with order of magnitude 100,000 varies very little. Accordingly to it, this value was chosen for the experiments.

Four cache replacement policies were considered: LRU, LFU-Aging, GD-Size and LRU-Threshold. We evaluate the impact of these policies combined in the different levels of the hierarchy. For a comprehensive description of several replacement cache policies, see reference [8].

4.1 Workload Characteristics

This section presents the main characteristics of the logs used in the experiments. These logs were obtained from a Brazilian ISP[1], with a hierarchical caching system similar to the one presented in figure 1. We obtained logs of two machines of the first level of this hierarchy, which we call *Pop-1* and *Pop-2*. The main workload characteristics are presented in Table 1. The logs of the days Oct 16-17, 2001 were used to warm the caches whereas the measurements of entropy and hit ratio were obtained with logs of the days Oct 18-19, 2001.

Note that the number of different objects represents about 26% of the total number of requests in all four logs. From this percentage, about 69% are documents with only one reference (*1-timers*). Moreover, our workloads contain mostly small objects. As show in Table 1, the 3° quartile of the distribution of file sizes is under $3KB$. Nevertheless, some objects are relatively large for Web documents, which explains the coefficient of variation of the distributions of file sizes being relatively high.

[1] POP-MG provides Internet access to incorporated customers and university users.

5 Experimental Results

This section presents the results of the simulation of the hierarchy of caches illustrated in figure 1. The average entropy was measured in the points numbered in the figure, which are the points where we perceive the effect of filtering, aggregation and disaggregation. For each caching polices LRU, LFU-Aging and GD-Size used in the caches at first level of the hierarchy; we evaluated different caching policies in the second level cache. The hit ratio and average entropy are calculated for each cache. Individual caches are identified as *child R*, *child L* and *Parent* for the first and second levels, respectively. Figures 2, 3 and 4 show the hit ratio and average entropy as the cache size increases.

Comparing the effectiveness of the first-level cache with the second-level cache, one can see that the first-level caches always get higher hit ratio. Comparing the entropy of the streams of requests before the hierarchy of caches with the entropy after the first level of caches, we verify that this occurs because the filtering of the first level caches absorbs part of the locality of reference and generates a stream of requests with smaller concentration of popularity for the second-level cache. The larger the first level caches, the higher is the filtering effect perceived. Thus, in some cases, the hit ratio of the second level cache decreases with the increase of the cache size at the first level. Moreover, as discussed in [2], the cache hit ratio becomes stabilized and reaches its maximum value when the cache is able to store all distinct objects. In our experiments, the maximum hit ratio for the first and second levels of the hierarchy occurs when the cache size is approximately 4GB.

We next discuss the variations of locality of reference comparing the entropy as the stream of requests pass through the hierarchy. We verify that the filtering diminishes the popularity when we compare the entropy of points 1 and 2 with the entropy of points 3 and 4, and the entropy of point 5 with the entropy

Table 1. Workload Characteristics

Item	**Pop-1**	**Pop-2**	**Pop-1**	**Pop-2**
Start Date	10/16/01	10/16/01	10/18/01	10/18/01
Duration (# days)	2	2	2	2
# requests	882,639	908,317	902,998	919,541
Distinct objects	234,663	246,560	238,880	237,290
1-timers	161,646	173,796	164,011	164,878
Workload Size (MB)	3,865	4,220	3,974	4,213
Smallest object	0	0	0	0
Largest object (MB)	33.13	41.75	29.61	49.70
Average Size (KB)	4.48	4.76	4.51	4.69
1° Quartile (Bytes)	365	372	364	371
Median (Bytes)	757	746	1,392	778
3° Quartile (Bytes)	2,690	2,698	2,576	2,571
Coefficient of Variation	16.52	29.29	14.22	19.62
Average Entropy	14.64	14.32	13.78	13.92

of point 6. Moreover, we notice that the aggregation in point 5 diminishes the entropy, increasing the concentration of popularity. The entropy in points 3, 4, 5 and 6 grows until stabilizing as the cache sizes increases. This occurs when the caches are able to store all distinct objects, and the entropy tends to its upper bound, which indicates a sequence without popularity.

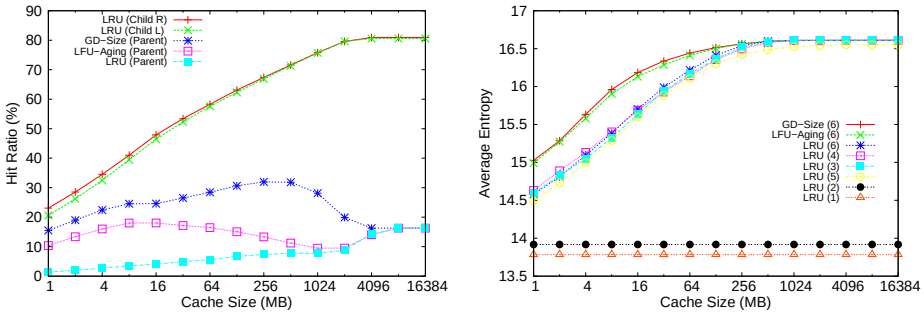

Fig. 2. LRU on the First Level - Hit Ratio and Average Entropy

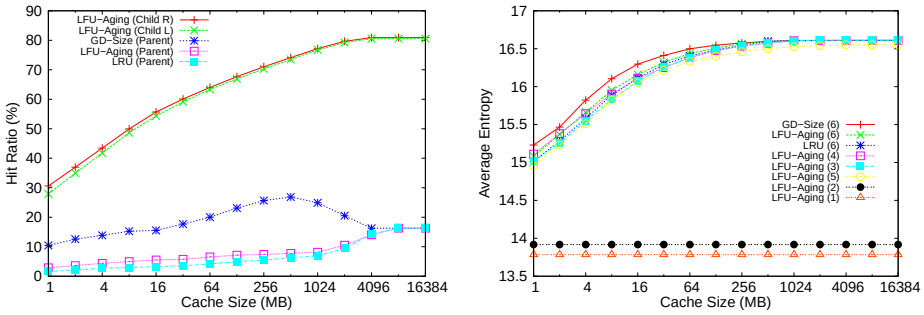

Fig. 3. LFU-Aging on the First Level - Hit Ratio and Average Entropy

Figure 5 shows the hit ratio and the average entropy when LRU-Threshold is used as the caching policy in the first level caches, storing just fewer files, with sizes smaller than $4KB$. Table 1 shows that this value is greater that the 3° quartile of the file size distribution for all logs, which indicates that most of objects can be stored in the first level, leaving only the largest objects to the second-level cache. Note that the sizes of the first level caches are large enough to hold all the objects smaller than $4KB$, the request stream that leaves these caches contains only references to objects smaller than $4KB$ that are 1-timers and to larger objects. Thus, the larger objects become relatively popular at the second-level cache, decreasing the entropy.

The relationship between entropy and hit ratio can be analyzed by observing the graphs in Figure 5. The reduction of the entropy in point 5, between the cache sizes 64 MB and 256 MB, has direct implication in the hit ratio of second-level cache. Until this point the LFU-Aging gets better hit ratio and, from this point on, GD-Size obtained the best result. This effect suggests that variations in the entropy can be used to dynamically configure caching policies in an hierarchy of caches. This is subject for future work.

In order to evaluate the performance of the hierarchy of caches as a whole, we simulated different configurations of caching policies. We consider the best configuration as the one that filters the concentration of popularity the most, i.e., the one which has the largest entropy in the stream of requests leaving the hierarchy. Figure 6 shows the final entropy and the hit ratio for some configurations of caching policies. The combination of LFU-Aging in the first-level caches and GD-Size in the second-level cache produced the best results, whereas the use of LRU in all caches performs the worst. Note that although the configuration with GD-Size in the two levels produced the best hit ratio, this configuration did not provide the best final entropy. This happens because these policies keep the smaller objects in the cache, thus increasing the hit ratio, but discarding bigger objects with some popularity. Therefore, this kind of policy does not act directly on the locality of reference.

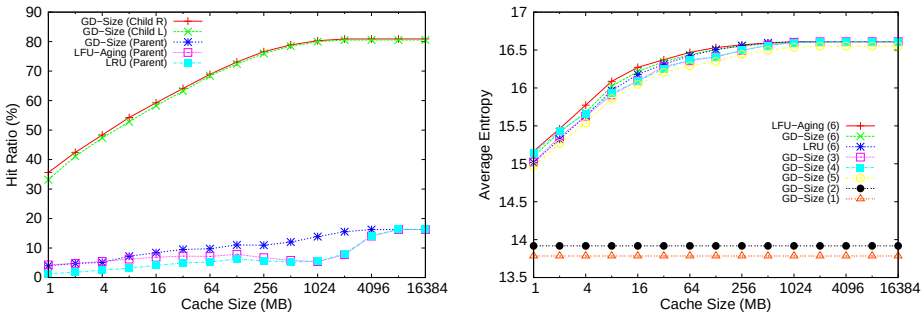

Fig. 4. GD-Size on the First Level - Hit Ratio and Average Entropy

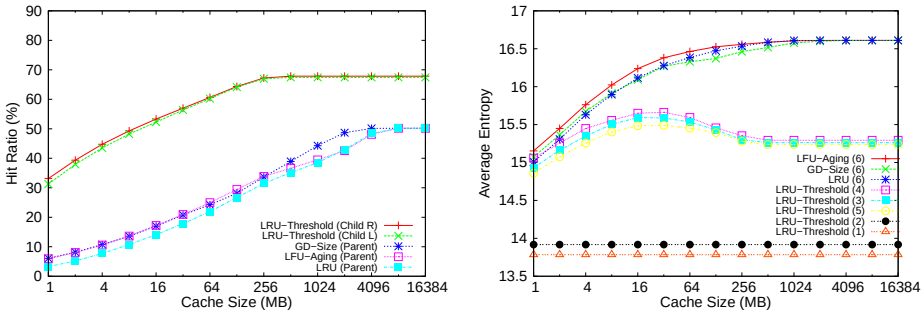

Fig. 5. LRU-Threshold on the First Level - Hit Ratio and Average Entropy

6 Conclusions and Future Work

We use the ADF (Aggregation, Disaggregation and Filtering) model and entropy to evaluate the effects that the locality of reference of a request streams suffer as they pass through a hierarchy of caches. The proposed metrics are able to capture online the locality of reference at any component of the Web hierarchy. We believe that the notion of locality can be used for operational decisions and thus, it can be useful to construct automated Web caching systems. Our results show how the transformations of aggregation, filtering and disaggregation act in the locality of reference and the impact of these operations into the performance of a hierarchy of caches. In general, filtering on the first-level caches is more effective than filtering on the second-level cache, since the request streams leaving the first-level caches have lower entropy. However, the aggregation of the outgoing first-level request streams decrease the entropy of the stream offered to the second-level cache, which provides an opportunity for a better hit ratio on that cache. Furthermore, our results show that heterogeneous configurations of caching policies take advantage of the reference locality.

Directions for future work include to explore dynamic and average entropy in proxy servers and to develop a model for hierarchical caching system in which the caching policies for the different levels of this hierarchy can dynamically be modified, based on variations of the entropy of the requests that arrive at the caching system.

References

1. V. Almeida, A. Bestavros, M. Crovella, and A. Oliveira. Characterizing Reference Locality in the WWW. In *Proc. of PDIS*, December 1996.
2. F. Benevenuto, F. Duarte, V. Almeida, and J. Almeida. Web Cache Replacement Policies: Properties, Limitations and Implications. In *Proc. of Latin American Web Congress*, November 2005.
3. L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker. Web Caching and Zipf-like Distributions: Evidence and Implications. In *Proc. of IEEE Infocom*, April 1999.

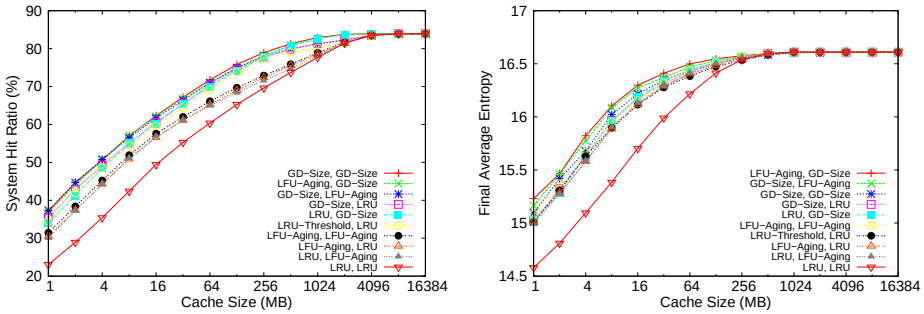

Fig. 6. Comparison among different configurations of the hierarchy of caches - Hit Ratio and Final Average Entropy

4. R. Fonseca, V. Almeida, and M. Crovella. Locality in a Web of Streams. *Communications of the ACM*, 48(1):82–88, January 2005.
5. R. Fonseca, V. Almeida, M. Crovella, and B. Abrahao. On the Intrinsic Locality Properties of Web Reference Streams. In *Proc. of IEEE Infocom*, May 2003.
6. A. Mahanti, D. Eager, and C. Williamson. Temporal Locality and its Impact on Web Proxy Cache Performance. *Performance Evaluation Journal: Special Issue on Internet Performance Modelling*, 42(2/3):187–203, September 2000.
7. S. Vanichpun and A. Makowski. Comparing Strength of Locality of Reference - Popularity, Majorization, and Some Folk Theorems. In *Proc. of IEEE Infocom*, March 2004.
8. J. Wang. A Survey of Web Caching Schemes for the Internet. *ACM Computer Communication Review*, 25(9):36–46, 1999.
9. D. Weikle, S. Mckee, and W. Wulf. Cache as Filters: A New Approach to Cache Analysis. In *Proc. of MASCOTS*, July 1998.
10. C. Williamson. On Filter Effects in Web Caching Hierarchies. *ACM Transactions on Internet Technology*, 2(1):47–77, February 2002.