On the Intrinsic Locality Properties of Web Reference Streams

Rodrigo Fonseca[‡] Virgílio Almeida[‡] Mark Crovella[§] Bruno Abrahao[‡]

[‡] Department of Computer Science
Federal University of Minas Gerais
Brazil

[§] Department of Computer Science
Boston University
USA

***Abstract*— There has been considerable work done in the study of Web reference streams: sequences of requests for Web objects. In particular, many studies have looked at the locality properties of such streams, because of the impact of locality on the design and performance of caching and prefetching systems. However, a general framework for understanding *why* reference streams exhibit given locality properties has not yet emerged. In this paper we take a first step in this direction. We propose a framework for describing how reference streams are transformed as they pass through the Internet, based on three operations: aggregation, disaggregation, and filtering. We also propose metrics to capture the temporal locality of reference streams in this framework. We argue that these metrics (marginal entropy and interreference coefficient of variation) are more natural and more useful than previously proposed metrics for temporal locality; and we show that these metrics provide insight into the nature of reference stream transformations in the Web.**

I. Introduction

Considerable effort has gone into the study of Web reference streams. Ever since the very first research on the Web appeared, a major line of study has focused on the properties of Web traces. This intense focus has been driven by the practical importance of understanding the nature of Web workloads, since it directly affects significant engineering and economic activity. This research has yielded important characterizations and insights. For example, the nature of Web request streams has informed the development of cache replacement policies [1], [2], inter-cache coordination protocols [3], and prefetching algorithms [4]. It has even become possible to contemplate the cost-optimal design of Web caches [5].

Nonetheless, the vast majority of the work so far on Web reference streams has dealt with individual streams *in isolation*, overlooking the fact that the Web is a system of clients, intermediaries, and servers, *each* of which may simultaneously emit or consume Web reference streams.

In this work we seek to advance this view of the Web as a system of agents emitting and consuming reference streams, moving to the next level of Web characterization and system design. We propose a shift in perspective, taking a stream-centric view of the Web which is long overdue. While performance engineering of individual agents (browsers, caches, servers) is fairly mature, an understanding of how to engineer *collections* of Web agents is almost nonexistent. Typical questions without good answers are: How should cache hierarchies be organized (how deep, in what relationship, and with how much local storage)? How does the choice of cache replacement policy depend on whether the cache is near to clients, near to servers, or distant from both? Even more fundamentally, what properties of Web reference streams are most useful in beginning to answer these questions?

We focus on how the characteristics of Web reference streams change as they pass through the Web, identifying three fundamental transformations that these are subject to: aggregation (merging streams), disaggregation (splitting streams), and filtering (removing some references from a stream). We want to develop an understanding of how each of these transformations affects the properties of Web request streams that are important for system design. In this paper, we focus on temporal locality as the first and most important property that affects the design of most Web agents. Our goal is to make quantitative statements about how temporal locality is affected by the processes of aggregation, disaggregation, and filtering in the Web. Our belief is that achieving this will bring us closer to an understanding of how to engineer entire collections of Web agents and answer some of the questions posed above.

We realized the collection of existing metrics for Web reference characterization was insufficient to this new task. Previous metrics used to assess temporal locality are tuned to the problem of studying individual agents in isolation, or are better suited to generating representative workloads, instead of explaining what causes the observed characteristics. In Section II, we introduce and motivate a set of metrics that effectively and accurately capture temporal locality, and yet are simple enough to lend themselves to reasoning about the three kinds of stream transformations. These metrics represent intrinsic characteristics of the requests streams, as distinct from those which appear due to the interaction of the stream with a particular system (e.g., temporal locality is instrinsic, whereas hit ratio is not). We describe our general framework for the analysis of Web systems in Section III, and in Section IV we show that our metrics are effective at capturing the essential properties of temporal locality in Web reference streams.

Then in Section V we put the pieces all together. Using a wide collection of traces collected at varying levels in the Web system, we show how the three transformations affect temporal locality as captured by our metrics. For example, we show evidence that the various transformations work together in the Web to make multilevel caching worthwhile. These

results suggest that our framework can allow a more principled understanding of how components of the Web interact, leading to better insight into the entire Web as a system.

II. Background and Related Work

The notion of temporal locality was first recognized by Denning [6] in his definition of the *working set,* which is the set of unique references contained within some fixed number of past references. This and other early work was in the domain of program memory references, as opposed to Web requests. Throughout this paper we refer simply to *object* references in situations where the distinction is not significant, and also use *virtual time*(following [7]), meaning that time is discrete and advances by one unit with each object reference.

Increased temporal locality generally improves cache performance. A demand-driven cache is designed to store some subset of the objects previously accessed; a good cache has the property that the objects held in the cache have a higher likelihood of being accessed in the near future than would a random collection of objects. When reference streams show temporal locality, then recency of reference is a useful metric upon which to base cache management policies.

Despite the extensive studies of temporal locality, it can sometimes be defined rather loosely, leading to differing approaches in its analysis. Two competing definitions tend to appear: (Definition 1) "An object just referenced has a *high* probability of being referenced in the near future" (*e.g.,* [8]); and (Definition 2) "An object just referenced has an *increased* probability of being referenced in the near future" (*e.g.,* [9]). While Definition 2 would seem to more accurately capture the spirit of temporal locality, Definition 1 better captures the importance of temporal locality for caching systems. For this reason, both definitions appear in the literature and the same term appears applied to both.

To make the distinction more precise, we will use a specific formalization for temporal locality. Consider a reference stream over a set of objects I, with $|I| = n$. We will use $p_i(k)$ to denote the probability that, following a reference to object i, the next reference to object i occurs within k references. We say that a particular object shows temporal locality of reference if there exists a $k > 0$ such that $p_i(k) > 1-(1-1/n)^k$. That is, object i shows temporal locality if there is some distance k over which two references to i are more likely than if references to i were independent with probability $1/n$.[1]

This formalization helps us see that temporal locality can arise in two ways: First, it can exist because an object is simply more popular than its peers, as when $p_i(k) > 1-(1-1/n)^k$ for all k. Second, it can exist when an object's references occur in a correlated manner, as when $p_i(k) > 1-(1-1/n)^k$ for only certain k.

Recently, a number of authors have focused on these two ways in which temporal locality can arise. The separation of temporal locality into these two effects was first suggested by Jin and Bestavros [10] and by Mahanti, Eager, and Williamson [9]. Jin and Bestavros termed these two effects *popularity* and *temporal locality*, while Mahanti, Eager, and Williamson use the term *concentration* instead of popularity; in this paper we consider them to be two kinds of temporal locality which we distinguish as *popularity* and *correlation*.

Probably the best-known characteristic of Web reference streams is their highly skewed popularity distributions, which are usually characterized by the term *Zipf's Law* [11]–[14]. Zipf's Law states that the popularity of the n^{th} most popular object is proportional to $1/n$. More generally, "Zipf-like" distributions have been found to approximate many Web reference streams well. In such a distribution:

$$P[O_n] \propto n^{-\alpha}$$

in which $P[O_n]$ is the probability of a reference to the n^{th} most popular object; typically, $\alpha \leq 1$.

The practical implication of Zipf-like distributions for reference streams is that most references are concentrated among a small fraction of all of the objects referenced. Returning to our definition of temporal locality above, this implies that for a small subset of objects, $p_i(k) \gg 1-(1-1/n)^k$ for all k. That is, Zipf's Law results in strong temporal locality because the probability of referencing certain objects is much greater than $1/n$.

Based on the near-ubiquity of Zipf-like distributions in the Web, many authors have used the value of α as a metric for capturing popularity skew [10], [14], [15]. However, for a number of reasons, we take a different approach. First, note that what is important in terms of temporal locality is the deviation of p_i from $1/n$ for some values of i, which is not directly captured by the Zipf exponent α. Rather, a direct measure of such deviation is available: *entropy.* The entropy of a random variable X taking on n possible values with probability p_i is simply:

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

The properties of $H(X)$ are exactly what we desire: it measures the deviation of X's distribution from the uniform distribution. It takes on its maximum value ($\log_2(n)$) in the case where all realizations of X are equally likely (*i.e.,* $p_i = 1/n,\ i = 1, \ldots, n$.) It takes on its minimum value (zero) in the case where only one observation can occur, that is, when the outcome is deterministic.

The second reason for preferring entropy over the Zipf exponent α is that real data often does not fit a Zipf-like distribution perfectly. As a result, measurement of the Zipf exponent can be subjective, as we will show in Section IV-A. The strength of the entropy metric is that it requires no underlying modeling assumption about the data, and so captures popularity skew equally well whether the trace adheres to a Zipf-like distribution or not.

[1] This formalizes the notion of Definition 1; in contrast, Definition 2 can be formalized (as discussed in [9]) as the property of decreasing probability of reference to an object with increasing time since its last reference.

Having identified an appropriate metric for capturing popularity, it remains to develop a metric for the correlation component of temporal locality.

Attempts to characterize correlation focus on the way in which references to a given object are separated by references to other objects. A widely used approach is the *stack distance* model [16]. For any given reference to object i, the corresponding stack distance is the number of unique references since the last reference to i (the first reference to i has undefined stack distance). The motivation for this metric derives from its relationship to the behavior of a cache managed with an LRU replacement policy, and also because it can be directly used to generate synthetic workloads [15]. Many studies have used stack distance to capture and characterize temporal locality in Web request streams [9], [13], [17], [18] and other authors have used the stack distance transformation as a tool for cache sizing [5] and workload generation.

However, stack distance does not fit our needs because it cannot distinguish the causes of temporal locality directly. As already noted, the authors in [9] consider both popularity and correlation, and capture the difference between these two kinds of temporal locality using a modified stack distance model. They propose that stack distance be normalized to factor out the effects of long-term popularity, allowing it to be characterized separately. Cherkasova and Ciardo [17] propose a similar approach based on normalizing stack distance.

The approach we take in this paper is to characterize correlation more directly, using a simpler and more intuitive metric than stack distance. Our approach starts from a simpler metric: the inter-reference distance. Instead of considering *unique* intervening references between two references to object i, we consider the total number of intervening references including those that appear multiple times. The advantage of this metric is that it does not intermingle the two kinds of temporal locality in the way that stack distance does. To see this, consider a sequence of symbols from a fixed alphabet. If we change the popularity of a symbol by replacing all occurrences of one symbol with another, then the stack distance properties of all symbols are potentially affected — even though the correlation properties of other symbols have not changed. This is not true for the inter-reference distance metric; it is purely a measure of correlation. A side, but important, benefit is that inter-reference distance is much faster and simpler to compute than stack distance.

Inter-reference distance was first used as a measure of temporal locality in [8]. More recently, inter-reference distance was used by Jin and Bestavros [10] as a measure of correlation; our work follows their lead in this regard. However we add to their approach in two ways: First, the measure of request correlation used in [10] is the distribution of inter-reference distance for *equally popular* objects, which may mix together objects that have varying distributions of inter-request times; we separate the inter-request distances for each object. Second, rather than fitting a line to the slope of the distribution of inter-request distances on log-log scale, which is fairly sensitive to noise, we summarize it in a simpler metric: coefficient of variation. This metric captures the essential properties of inter-request distance, as shown in Section IV-B.

After developing these natural and precise metrics for capturing temporal locality, we show how these metrics are affected by the transformations on request streams that commonly occur in the Web: stream aggregation, disaggregation, and filtering. The first two have not been extensively studied before. The third transformation (the stream filtering effects of caches) has recently received some attention. Weikle et al. [19] introduce the view of caches as filters, and compare properties of incoming and outgoing streams of references, in the context of program memory references. In the context of Web caching, Mahanti, Williamson and Eager [9] study how temporal locality changes at different levels in the caching hierarchy. They show that the concentration of references tends to diminish, and the tail of the zipf distribution increases, as one goes up the caching hierarchy. This effect is also noted and characterized in [20], [21]. These papers are consistent with a subset of our results; we put these effects (and others) into a larger framework. For example, while [20] examines filtering effects of caches, it does not separate the effects of filtering on the two kinds of temporal locality, nor does it consider transformations other than filtering.

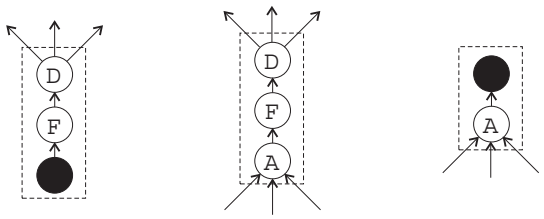
III. A Framework for Analyzing Web Request Streams

One of our main goals is to establish a framework for analyzing the Web as a system of agents operating on streams. Our motivation is to be able to isolate the different phenomena that affect reference streams, so as to gain insight into the characteristics of streams at different points in this topology.

Traffic on the Web can be seen as a sequence of requests originated by clients that flow "upward" in some sense, and responses to these requests that flow "downward". The different streams of requests, although presenting different characteristics, all share some commonalities in the way the requests flow and interact, dictated both by the HTTP protocol and by the topology of the Web system.

Our abstraction views the topology of the Web system as a graph, in which the nodes represent points where the streams may be altered, and the edges are paths connecting these points. Some nodes in our graph are the points at which the application level surfaces, *i.e.,* the endpoints of the TCP connections through which web requests are transported. However, nodes can also be specific portions of application software. For example within a client browser there is a stream of requests that is generated by user actions, which is passed to the browser cache. The request stream that is sent into the network originates from the miss stream of this cache. Thus, the browser cache can be represented by a node that alters the original request stream by filtering it.

The nodes in our graph are of three different types, depending on what their effect is on the request streams: Aggregators (A), Disaggregators (D) and Filters (F); different components of the Web topology may be represented by combinations of these three kinds of nodes. There are also endpoint nodes,



(a) Client (b) Proxy Cache (c) End Server

Fig. 1. Abstractions of typical components of the Web topology in the ADF graph. The node types are Aggregator (A), Disaggregator (D), Filter (F), and End Nodes (dark), and correspond to points in the topology at which the Web request streams can be altered.

which can generate and absorb requests; these are clients and servers, respectively.

The three types of nodes correspond to three phenomena affecting streams. As mentioned above, the request stream that a user generates goes through a filter node (F) (the browser cache). After this, this same request stream is split into different TCP 'pipes', that go to different destinations. This can be abstracted by a disaggregator (D) node. Disaggregation will most commonly happen when requests are sent to different destinations, but may be artificially induced by content. For example, a page served by a content provider which has a contract with a CDN may have requests directed to varying locations. Figure 1(a) shows how a client just described might be represented in our model.

Another important example is that of a proxy cache, which is often located at some intermediate point of the topology, between many clients and many servers. When requests are received by a common cache server through different TCP connections they are aggregated in a single stream, so that the cache subsystem itself sees a single stream. This can be represented by an aggregator node in our graph. This aggregated stream is then processed by the cache, and the miss stream that leaves the cache is the result of a filtering operation by the cache. The next node is a disaggregator node, from which several edges leave, headed towards different servers. Of course, some of these may be other proxy servers, as well as end servers or content providers. Figure 1(b) shows one possible configuration of a proxy cache server in our graph.

A content provider, or an end point server, has similar components. Right before the processing by the Web server, which may even be a cluster of servers, there occurs an aggregation operation, and several streams coming upward are multiplexed into a single stream. Of course the details of this aggregation would depend on factors such as load balancing policies, which might be represented by several independent aggregators, or by a sequence of aggregators and disaggregators. A possible representation for a simple server is shown in Figure 1(c).

Having categorized Web stream transformations into these three kinds, we can begin to ask questions about why streams show particular locality properties. That is, we can ask: how does locality change when streams are aggregated? disaggregated? filtered? In order to approach these questions we need precise metrics for locality that are amenable to this style of analysis. In the next section we describe those metrics, and in the following section we use those metrics to provide some initial answers to these questions.

IV. Measuring Temporal Locality

To move to a view of the Web as a collection of reference streams, we need to define quantitative measures of temporal locality that are suitable for use in this framework.

Because we are analyzing streams abstractly, we avoid using metrics that are dependent on the properties of any particular system. This is in contrast to traditional metrics that have been applied to caching systems, (*e.g.,* hit ratio and byte hit ratio [22]) which are strongly dependent on the parameters of the cache itself (such as cache size and replacement policies) and only indirectly reflect the intrinsic properties of the stream.

Furthermore, although we are interested in analyzing these intrinsic properties of the streams, it is only possible to estimate them based on particular finite stream realizations — logs that record request sequences (*e.g.,* client, proxy, or server logs). Therefore we need to pay careful attention to issues of normalization for artifacts such as the particular number of requests or distinct objects in the log.

As described in Section II, temporal locality can be decomposed into two effects: popularity and correlation. In the following two subsections we define and motivate metrics we use for measuring these two effects; we discuss normalization issues; and we show that the resulting metrics are more general and robust than previously used metrics for temporal locality. The logs we use for some of the validations are described in Section V.

A. Measuring Popularity: Entropy

In this section we define our metric for measuring the skew (degree of imbalance) in the relative popularity of different objects in a request stream, and demonstrate its strengths.

1) Definition: To measure the degree of imbalance in the popularity of objects, we use the entropy of the request stream [23], [24]. That is, the stream is treated as independent samples of a random variable X, and we are concerned with the entropy of X, denoted $H(X)$.

In estimating $H(X)$ from a given log, we view the requests in the log as a sequence of symbols, which are the requested objects; we approximate the probability p_i of a given object i being referenced as the number of times it appears in the log, divided by the total references in the log. We thus obtain an empirical probability distribution over the set of objects in the log. Then the entropy $H(X)$ is defined as in Equation (1). Note that $H(X)$ only depends on the probabilities of occurrence of the different objects, and not on the relative order in which they are occur.

As defined in Equation 1, $H(X)$ also depends on the number of distinct objects that are referenced in the log. In

particular, the upper bound on entropy is given by $H_0(N) = \log_2(N)$, and is attained when each object is equally likely to be referenced. It has been shown that the number of distinct references in a segment of a log increases with the size of the segment, even for very large logs [25]. Thus to be able to compare logs with different number of distinct references present, it is important to normalize the entropy measure. The appropriate normalization is based on the largest possible value of $H(X)$, namely $H_0(N)$. Therefore the metric for popularity we will use is the normalized entropy:

$$H^n = H(X)/H_0(N) \tag{2}$$

where N is the number of distinct references in the log.

Finally, we note that in some of our later results, instead of working with H^n, we use the following transformation of H^n:

$$H^s = -\log_{10}(1 - H^n) \tag{3}$$

We use H^s (which we called "scaled" normalized entropy) because H^n can often be quite close to 1, making it hard to distinguish on plots.

2) Validation: In this section we show that normalized entropy accurately captures the popularity component of temporal locality by relating normalized entropy to two commonly used measures of this property: hit ratio and Zipf exponent α. As discussed in Section II, α is the most commonly used measure of the degree of imbalance in popularity of Web references. Here we show three key facts:

1) For any given number of unique references N, if reference popularity precisely follows a Zipf-like distribution, then there is a one-to-one relationship between the Zipf exponent α and $H(X)$;
2) In some cases, reference popularity does not precisely follow a Zipf-like distribution, making the estimation of α error-prone, but having no such effect on $H(X)$; and
3) Across a large set of traces and cache sizes, normalized entropy is strongly correlated with hit ratio.

Our first point is that in the case where popularity in fact follows a Zipf-like distribution, for a trace with N unique references, $H(X)$ has a one-to-one relationship with α. This means that, given $H(X)$ or H^n, one can determine the corresponding α — so there is no need to additionally measure it. The fact can be seen as follows: Let $S_{\alpha,N} = \sum_{i=1}^{N} i^{-\alpha}$. Then if a trace obeys a Zipf-like distribution, $p_i = \frac{1}{S} i^{-\alpha}$, so:

$$\begin{aligned} H_{zipf}^{\alpha,N}(X) &= \sum_i -p_i \log_2 p_i \\ &= \log_2 S + \frac{\alpha}{S} \sum_i i^{-\alpha} \log_2 i \end{aligned} \tag{4}$$

The relationship between $H(X)$ and α is intuitively clear: as α approaches 0 the popularity distribution becomes more uniform, and the entropy tends to $log_2 N$ (as can be seen by $\lim_{\alpha \to 0} H_{zipf}^{\alpha,N} = \log_2 N$). On the other hand, when α grows the popularity distribution becomes more imbalanced, and in the limit only one object is referenced so the entropy is 0 ($\lim_{\alpha \to \infty} H_{zipf}^{\alpha,N} = 0$). This relationship is illustrated in Figure 2. Most of the reported estimates of α lie in the range between 0.6 and 1 [14]. In this range, the Figure shows that normalized entropy H^n typically lies between approximately 1/2 and 1.

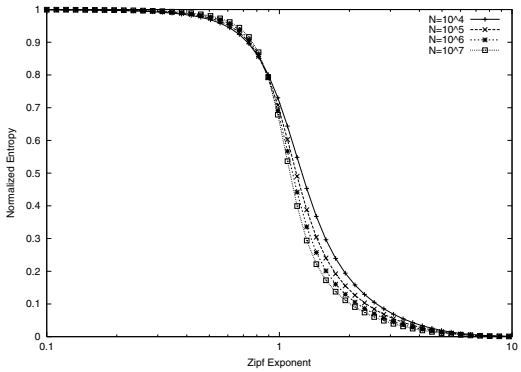


Fig. 2. Normalized entropy of ideal streams with Zipf-like popularity distribution of references, as a function of the Zipf exponent α.

Our next point concerns the lack of generality and accuracy inherent in using the Zipf-like distribution as a model.

In Figure 3 we show, for one of the traces we studied (others, not shown, are similar), estimates of the Zipf exponent using two different regression techniques (lines marked 'zl' and 'ze'). The figure shows how reasonable regression procedures can produce widely varying results. The 'zl' line was determined by a linear fit on the logarithm of the references versus rank data; the 'ze' lines was obtained from a non-linear fit on the original data.

We can see that the linear fit ('zl') emphasizes the points in the tail of the distribution, while the non-linear fit on the original data ('ze') emphasizes the highest ranked objects. It is a matter of judgement to determine which points to use and which procedure to adopt, in order to correctly estimate the exponent.

In fact, a third estimate of α is possible: that based on inverting Equation (4). By first calculating $H(X)$, and then obtaining the corresponding α value, we obtain an estimate of α that is independent of which points we use to fit the actual popularity distribution; these lines are marked 'zh' in Figure 3. Thus we argue that $H(X)$ is a fundamentally more robust metric than is α.

Cache size (%)	Correlation H^n X HR	
	Scrambled Logs	Original Logs
1	-0.72	-0.61
2	-0.83	-0.77
5	-0.84	-0.86
10	-0.81	-0.84
14	-0.79	-0.81
20	-0.78	-0.79
29	-0.76	-0.76
50	-0.74	-0.74

TABLE I

CORRELATION COEFFICIENT BETWEEN H^n AND HIT RATIO.

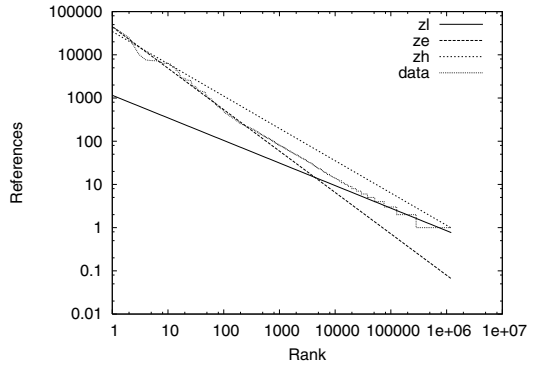


Fig. 3. Determination of the Zipf parameter using the entropy ('zh' curve) and two standard regression techniques for the NLANR SD log.

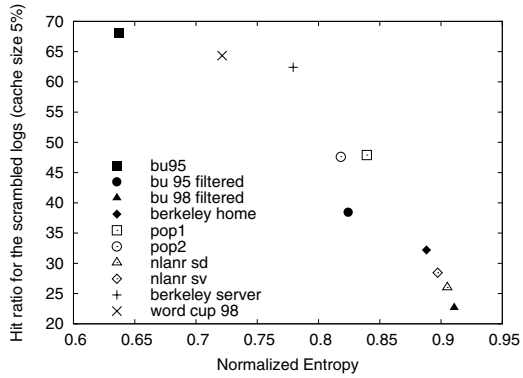


Fig. 4. Hit ratio for the scrambled versions of the logs versus the normalized entropy of the logs. The cache uses LRU replacement policy, and has the capacity to hold 5% of the total objects in each log.

Our third point is that H^n has a strong effect on hit ratio which suggests that it provides a useful view of locality. To demonstrate this we show the results of cache simulations. We simulated an LRU cache and did not consider the size of the objects referenced, *i.e.,* we considered all objects to be the same size. While this is not an accurate measure of true hit ratio considering object sizes, it can provide valuable insight into the utility of the entropy metric.

Figure 4 shows the results when caches are sized to hold 5% of the number of unique references in each log. Logs were scrambled to remove all temporal correlations, leaving only the popularity component of the locality. The figure shows the clear and strong trend of decreasing hit ratio with increasing entropy. A wider range of results are summarized in Table I. This table shows, for a range of cache sizes, the correlation coefficient (R^2) relating the normalized entropy to the hit ratio obtained in the caching simulation for that size. The table shows that the correlation between hit ratio and normalized entropy is strong across a wide range of cache sizes.

In fact, this result is supported by theory in [26], which establishes that entropy can be used to establish bounds on the miss rate of an optimal caching algorithm for a discrete memoryless source with finite alphabet. It is important to note however that they use the entropy rate of the source of the request stream, and that the measure of entropy that we use here is a first order approximation of the source entropy rate. For details, we refer the reader to [23], [26].

Finally, Table I also shows results for the original, non-scrambled versions of logs. It is interesting to note that entropy bears a strong (negative) correlation with the hit ratio for the original logs, especially for larger cache sizes. This may be understood because the larger the cache, the less important the impact of temporal correlation is on the cache performance.

B. Measuring Correlation: the Coefficient of Variation

Turning to measures of correlation, in this section we describe our metric for the correlation component of temporal locality, and and show its utility.

If all accesses to objects in a stream were completely uncorrelated, we could model the request generating process by the Independent Reference Model [7] (IRM). In the IRM, each object has associated a probability of being referenced, and each reference is independent of any other reference. In this model the inter arrival time (IAT) (measured in number of references) follows a geometric distribution, which is a memoryless distribution. For each object i with reference probability p_i, the mean of its associated IAT distribution is given by $1/p_i$.

If we form a random permutation of a log of references, we expect the IAT of each object to follow a geometric distribution, with mean determined by the relative frequency of appearance of the object. In the original log, however, if there is temporal correlation between the references to the same object, we expect to see deviation from this memoryless behavior.

As an initial example, to distinguish the presence of correlation from that of popularity, we proceed as the authors in [10], grouping equally popular objects and conducting separate analysis of these groups. In the absence of temporal correlation, we would have the IRM, as discussed above, and the distribution of the IAT's for any given object would depend solely on the probability of accesses to that object. Thus, equally popular objects would have the same IAT distribution.

We thus examined the IAT distribution for groups of objects with the same number of accesses, and obtained similar results for all such groups. As an example, we show in Figure 5 the IAT distribution for one of these groups, that of all objects with 8 references each.

We plot the cumulative distribution of the IAT for the objects in the original log (plot (a)), and for a scrambled version of the same trace (plot (b)). Figure 5(a) shows a strong tendency of the IATs to cluster around shorter values, and that this tendency is not present in the same plot for the scrambled trace (b). We also show curves labelled 'IRM', which are the corresponding geometric distribution. In plot (b) we see how close the scrambled version approaches the curve for the geometric distribution. The plots for all other frequencies show very similar behavior, and these results confirm those obtained (in a slightly different fashion) in [10].

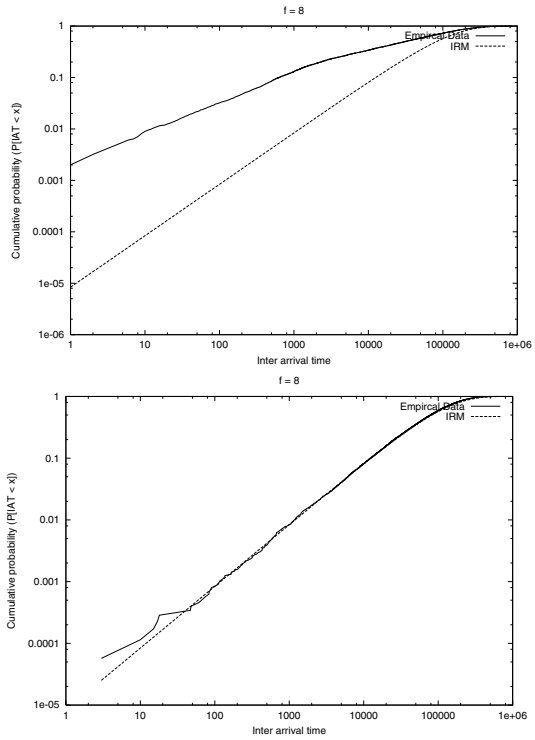


Fig. 5. IAT distribution for objects with frequency 8, for the original (top) and scrambled (bottom) versions of the NLANR SV log.

1) Definition: Motivated by the fact that temporal correlation for the accesses to the same object should cause a deviation from the geometric distribution, we introduce a metric that is very sensitive to this deviation. The *coefficient of variation* of a distribution is its standard deviation σ divided by its mean μ. Coefficient of variation (CV) is widely used in different settings, and is a simple measure of relative dispersion of a distribution. This metric is dimensionless and is simple to calculate and understand.

CV is also a convenient measure to use here because it has a natural reference point for the geometric distribution. Given a geometric distribution with mean given by $\mu = \frac{1}{p}$ and variance given by $\sigma^2 = (1-p)/p^2$, the CV is given simply by $\sqrt{1-p}$. In Web reference streams, in which even the most heavily accessed objects have a very low probability of reference (generally much less than 1%), the expected CV, in the case of no temporal correlation, is very close to 1. Thus, in this setting, CV values close to unity are associated with distributions close to the IRM, and thus little or no temporal correlation, while values larger than one represent a distribution with large relative variance.

In order to use the IAT-CV as a measure of correlation, we must decide how to apply it across the unique set of references in a trace. In [10] the authors form IAT distributions over the set of all objects with k references. This has the effect of mixing together objects with possibly varying properties. In contrast we wish to consider each object separately and so form the IAT-CV for each unique reference in the trace. Furthermore, unlike [10], using IAT-CV as a summary of the distribution in this way does not require any assumption about distributional shape nor does it require any manual curve-fitting.

Each unique object in a trace has its own IAT-CV. In order to summarize an entire trace we must combine these individual values. The method we use is based on the *per-reference* IAT-CV. That is, since we are concerned with temporal locality as it impacts caching systems, we seek a metric that is weighted on a per-reference basis (rather than, say, a per-unique-reference basis).

Interestingly, the distribution of per-reference IAT-CV shows a long tail, making the mean of this quantity an unstable metric. We find experimentally that the long tail of this distribution results in a mean IAT-CV that grows with the length of the trace considered. Since we want to normalize our metrics to be independent of trace length, we need a statistic that is insensitive to trace length. For this reason we use the distributional median, which is a very robust metric even for long-tailed distributions.

To sum up, the metric for correlation we use is calculated as follows. For each object in the trace, calculate its IAT-CV (the standard deviation of its IAT over its mean IAT). Now form the set consisting of the IAT-CV for each individual reference in the trace; an object with n references in the trace will have n copies of its IAT-CV in this set. The final metric is the median value of this set.

2) Validation: The presence of correlations in a reference stream can be measured by its effect on cache hit ratio. When a trace is scrambled, the resulting hit ratio tends to decrease due to the removal of correlation. This difference may be used to gauge the utility of our IAT-CV metric. In Figure 6 we show the relationship between the difference in hit ratio due to scrambling, and the IAT-CV. In this case all caches were sized to hold 5% of the unique references in each trace. The figure shows the clear relationship between improvement in hit ratio and IAT-CV. Larger IAT-CVs are indicative of stronger contributions of correlation to the overall hit ratio of the trace in a cache.

These results are generalized to a range of cache sizes in Table II. The table shows that even for large cache sizes (capable of holding half of the unique references in a trace), that the improvement in hit ratio is correlated with IAT-CV; and for small caches, this correlation is quite strong.

V. Effects of the Transformations on Request Streams

In this section we analyze the effects of the three ADF transformations on the temporal locality properties of streams. We do this through a set of experiments on logs taken from different points of the topology, using the metrics introduced in Section IV. A short description of these logs is given in Tables III and IV. All of these, except for the POP1 and POP2 traces, are publicly available and can be found in [27] or [28]. They are more thoroughly described in [29].

Cache size (%)	Correlation CV X HR diff.
1	0.78
2	0.72
5	0.65
10	0.53
14	0.44
20	0.38
29	0.31
50	0.32

TABLE II

CORRELATION COEFFICIENT BETWEEN CV AND THE HR DIFFERENCE (ORIGINAL - SCRAMBLED), FOR DIFFERENT CACHE SIZES.

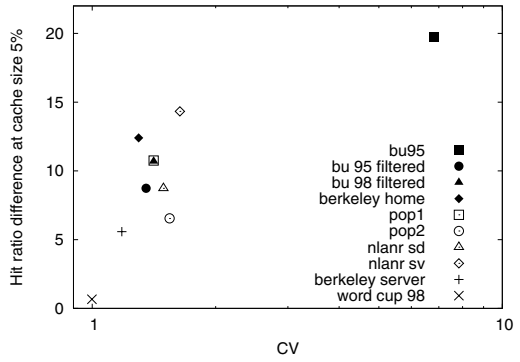


Fig. 6. Difference in hit ratio between the original and the scrambled logs, for an LRU cache with relative size of 5%.

Name	Short Description	Period
Client, before browser cache		
bu95	'95 Boston University[2]	01/01 - 02/28/95
Client Proxy, after browser cache		
bu95flt	'95 Boston University	01/01 - 02/28/95
bu98ftl	'98 Boston University	04/06 - 05/21/98
bk-homeIP	Berkeley Home IP Service	11/06 - 11/09/96
Network Proxy		
pop-1	POP-MG Level 1 Cache	10/18 - 10/19/01
pop-2	POP-MG Level 2 Cache	10/18 - 10/19/01
nlanr-sd	NLANR SD Root Cache	11/13 - 11/19/01
nlanr-sv	NLANR SV Root Cache	11/13 - 11/19/01
End Server		
bk-server	Berkeley CS Dept.	12/01 - 12/31/01
wc	'98 World Cup Web Site	05/29/98

TABLE III

HIGH LEVEL DESCRIPTION OF THE LOGS USED

Name	Requests	Objects	% 1-Timers
bu95	558,263	48,532	43.64
bu95flt	128,077	47,502	72.14
bu98flt	67,629	35,646	42.02
bk-homeIP	1,703,835	600,940	66.18
pop-1	1,949,490	464,795	69.53
pop-2	2,308,411	734,015	76.26
nlanr-sd	3,950,198	1,785,884	18.17
nlanr-sv	5,357,077	1,544,956	35.38
bk-server	6,011,564	268,018	1.69
wc	2,223,141	4,829	0.04

TABLE IV

SOME IMPORTANT STATISTICS OF THE LOGS USED

We broadly divide the logs we used into 3 groups, according to their location in the topology of the Web: client, proxy, and server logs, and this distinction is useful in some of the discussions that follow.

A. Filtering

Trace driven simulations with empirical traces collected at different points in the Web system are used to assess the effects of filtering on reference streams. Because the goal of the experiment is not to analyze replacement policies, but to understand the effects of caching on reference streams, the simulations assume a simple LRU policy.

When analyzing the filtering transformation, we want to understand how entropy and CV capture the temporal locality properties of the output stream. The two following questions help us to understand the effects of this transformation: 1) how do the locality metrics of the output stream vary as a function of the cache size? and 2) how do the filtering effects vary according to the topological position of the caching in the Web?

Intuitively, we expect that filtering absorbs part of the temporal locality of a reference stream and generates a miss stream consisting of evenly distributed references to fairly popular objects. The graph of Figure 7 shows the variation of the normalized entropy as a function of the cache size. In the graph, note that points corresponding to cache size equals to 0 refer to the entropy of the input stream. The figure shows that the entropy of the output stream (i.e., miss stream) increases

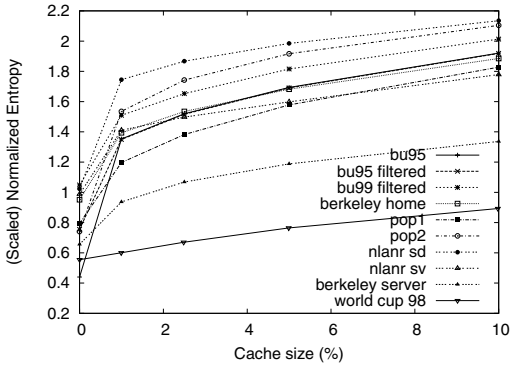


Fig. 7. (Scaled) normalized entropy of the miss stream versus cache size. Cache size is measured as a fraction of the total number of objects in the log. The point for cache size of 0 corresponds to the original log.

with the cache size, indicating that more popular references are eliminated from the stream as we increase the size of the cache, making the object distribution of the output stream more uniform.

The figure also shows that the deeper the cache in the Web system, the lower the magnitude of the entropy difference between input and output streams. The explanation for that stems from the different percentage of one-timers in the reference streams. As indicated in Table IV, one-timers in the two traces near to clients constitute 42% and 66% of references. In the World Cup and Berkeley servers logs, the one-timers represent

0.04% and 1.69%. In a popularity versus rank plot, the curves for the server logs would have a concentration of points at the top left portion of the curve while the client logs would have a concentration of points at the tail of the curve. Williamson [20] shows graphically that the primary impact of filtering is to truncate and flatten the top left portion of popularity curves. Therefore, entropy variation is magnified by the presence of a large percentage of one-timers. Williamson [20] also shows that deeper levels of caching produce little change in the object popularity profile, which agrees with the results shown here.

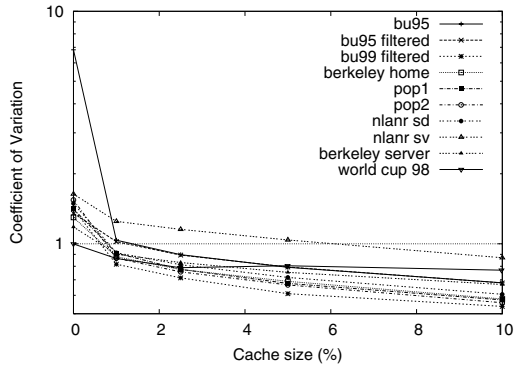


Fig. 8. CV of the miss stream versus cache size.

Figure 8 displays the IAT-CV as a function of cache size. As the cache size increases, more repetitions are eliminated from the output stream, decreasing the temporal correlation of the miss stream. As a matter of fact, the miss stream should exhibit a low coefficient of variation, for the temporal correlations are almost eliminated by the LRU policy of the caching. For larger cache sizes, since we use a cache simulator that only has capacity misses, and no expiration misses, the number of requests in the miss streams begins to diminish quickly, and so does the number of repetitions. This makes the number of objects that actually contribute to the value of IAT-CV to also diminish, and the metric loses precision for these larger caches.

In summary, we find that the effects of caching are to remove both sources of temporal locality from reference streams. This suggests that the output stream from a cache would typically be a poor candidate for sending to another cache, since the subsequent "downstream" cache would observe little temporal locality in its input stream. Nonetheless, we observe that multilevel caching is common in the Web, from client caches to proxy caches to server accelerators (caches in front of servers). Why are such multilevel caching schemes effective? The answer is provided by turning again to the ADF framework, which we do in the next section.

B. Aggregation and Disaggregation

In order to analyze the effects of the aggregation and disaggregation transformations on the streams, we analyze different logs by separating them into sub-logs corresponding, in turn, to the sources of the requests, and to the destinations of the requests. We then study the distribution of H^n and IAT-CV across these sub-logs. Note that we do not use any filtering between the two transformations, for we want to isolate their individual effects. The plots in Figure 9 show the cumulative distribution *per reference*, i.e., weighted by the number of requests in each sub-log, of H^n and the CV, for a sample log. The full vertical lines in each plot shows the value of the corresponding metric in the full log.

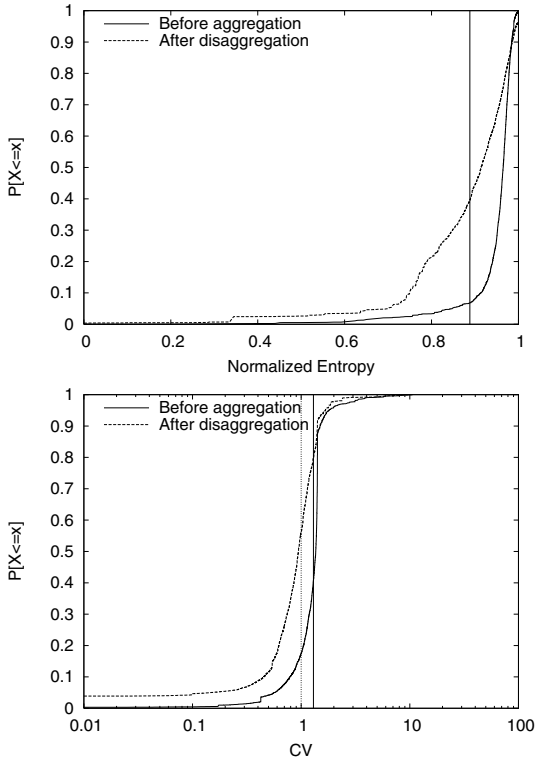


Fig. 9. Cumulative distributions of H^n and cv for the Berkeley Home IP log, for the aggregation and disaggregation sub-logs. The full vertical line is the respective value for the full log.

We first examine the results for normalized entropy. Across all logs we studied, the percentage of requests from incoming sub-logs with H^n greater than that of the full log is consistently above 50%. This shows that aggregation tends to *decrease* the normalized entropy of its output stream compared to its input. That is, aggregation acts to increase the popularity component of temporal locality. This provides an answer to the question posed at the end of the last section: higher level caches tend to be effective despite lower level caching because they generally involve considerable stream aggregation (as shown in Figure 1(b)). This is an example of how we can use the ADF framework to enable better engineering of the Web system, since we can decide to place caches at locations in which high levels of aggregation occur.

For the case of the disaggregation, although the median of the distribution is close to the reference value for the full log, one can notice a significant increase in the number of requests belonging to outgoing sub-streams with lower normalized entropy. This is also intuitive, because the streams that servers

receive have a more limited set of objects, and the requests can be drawn from several different incoming streams, the repetitions of which were not filtered together before.

Concerning the IAT-CV, we have found for all logs we studied that the distribution per reference of the IAT-CV decreases when comparing the incoming sub-streams for the aggregation to the outgoing sub-streams for the disaggregation. This can be seen for the second plot in Figure 9. This finding is in line with the intuitive notion that the temporal correlations should decrease when aggregating, because the separate processes that generate the requests (for example, two different users) are most likely independent time series. The same observation applies to the disaggregation transformation.

Thus we find that aggregation and disaggregation shed considerable light on the Web system: both aggregation and disaggregation tend to increase the popularity component of temporal locality, while tending to somewhat decrease the correlation component of temporal locality. These conclusions are supported by looking at the properties of traces at different levels in the hierarchy, which we do in the next section.

C. Locality properties in different points of the topology

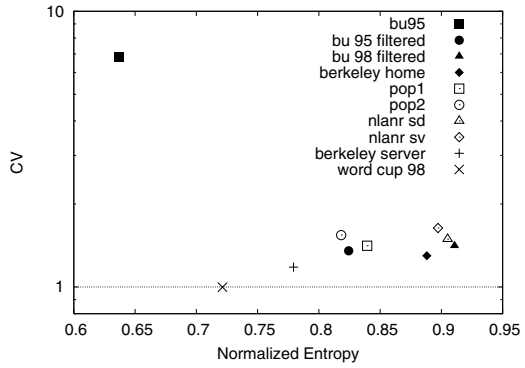


Fig. 10. CV versus Normalized Entropy for logs studied. Logs from locations close to clients are shown with filled symbols, logs from proxy servers are shown with hollow symbols, and the logs from servers are shown with line drawn symbols.

Having studied the effects of the transformations individually, we now present, in Figure 10, results combining both metrics for each of the ten logs we studied, which come from different points in the topology. We can readily notice three different groups of points. The first is that composed of one point, 'bu95'. This is the only trace that is collected before browser caches, and has not been filtered in any way. It presents a high degree of temporal correlation, that comes from the correlations in user surfing patterns, and also the lowest relative entropy, since no repetitions have been filtered out yet. Then we move to the 'proxy region', in which the streams present high H^n and a much lower CV. These characteristics come from the fact that these proxies generally receive streams that have been filtered by lower level caches. It is interesting to notice that even proxies that are close to clients exhibit these characteristics. The third region in the plot is the 'servers' region. These present the lowest temporal correlation, and also lower entropies. This is in line with our findings that aggregation and disaggregation both increase popularity imbalance and that all transformations tend to decrease temporal correlation.

These findings show that the correlation and popularity components of locality behave differently as streams pass through the Web system; while popularity imbalance can rise or fall as a result of stream transformations, correlation seems to be intrinsically generated by clients and generally declines as streams are transformed.

VI. Conclusions and Future Work

In this paper we have presented a shift in perspective in Web characterization, focusing on properties of request streams, and how these are affected by aggregation, disaggregation, and filtering transformations.

We have proposed new metrics to capture the two causes of temporal locality: popularity and correlation. Entropy was defined as a natural metric for measuring the skew in the relative popularity of different objects in a request stream, and the Coefficient of Variation of the IAT distribution was used as a metric for temporal correlation, motivated by the fact that the presence of such a correlation between accesses to the same object should cause a deviation from the geometric distribution. We validated these metrics using a wide collection of logs from different points in the Web topology, and showed that they are intuitive and effective.

We then showed how these metrics can give valuable insights when applied to the transformation framework. For example, they yield an important observation that concerns the effectiveness of caching hierarchies: while caches themselves *decrease* temporal locality, aggregation and disaggregation can *increase* temporal locality. These observations provide guidance for overall design of the Web system: *e.g.,* in suggesting that stream aggregation (perhaps more than network topology) is a key factor in optimizing cache placement.

This point is refined by our separate measurement of the two components of locality. While the popularity component can sometimes increase (leading to the effect just described), all three transformations were found to diminish the correlation component, and this effect was confirmed by the consistent decline in correlation found in streams moving up the hierarchy from clients toward servers.

Looking to the future, we believe that in some cases, bounds on the values of the resulting streams can be determined, given the parameters of the input streams and transformation node. We are currently investigating analytical frameworks for studying cache system performance in light of the properties discussed in this paper. We also see these metrics as important tools for understanding and better designing the Web as a system of multiple interacting agents that create, transform and absorb streams of request.

Acknowledgments

This work was partially supported by NSF grant ANI-9986397 and by a grant from Sprint Laboratories. Rodrigo was supported by CNPq grant 200013/3149-5.

REFERENCES

[1] J. Wang, "A survey of Web caching schemes for the Internet," *ACM Computer Communication Review*, vol. 25, no. 9, pp. 36–46, 1999. [Online]. Available: citeseer.nj.nec.com/wang99survey.html

[2] P. Cao and S. Irani, "Cost-aware WWW proxy caching algorithms," in *Proceedings of the 1997 Usenix Symposium on Internet Technologies and Systems (USITS-97)*, Monterey, CA, 1997. [Online]. Available: citeseer.nj.nec.com/cao97costaware.html

[3] L. Fan, P. Cao, J. Almeida, and A. Z. Broder, "Summary cache: a scalable wide-area Web cache sharing protocol," *IEEE / ACM Transactions on Networking*, vol. 8, no. 3, pp. 281–293, 2000. [Online]. Available: citeseer.nj.nec.com/fan98summary.html

[4] A. Bestavros, "Using speculation to reduce server load and service time on the www," in *Proceedings of CIKM'95*, Baltimore, Maryland, November 1995.

[5] T. Kelly and D. Reeves, "Optimal Web cache sizing: Scalable methods for exact solutions," in *Proceedings of the 5th Web Caching Workshop*, May 2000.

[6] P. J. Denning, "The working set model for program behavior," *Communications of the ACM*, vol. 11, no. 5, pp. 323–333, May 1968.

[7] P. J. Denning and S. C. Schwartz, "Properties of the working-set model," *Communications of the ACM*, vol. 15, no. 3, pp. 191–198, Mar. 1972.

[8] V. Phalke and B. Gopinath, "An interreference gap model for temporal locality in program behavior," in *Proceedings of the 1995 ACM SIGMETRICS Conference*, 1995, pp. 291–300.

[9] 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*, vol. 42, no. 2/3, pp. 187–203, Sept. 2000.

[10] S. Jin and A. Bestavros, "Sources and Characteristics of Web Temporal Locality," in *Proceedings of the 8th MASCOTS*. IEEE Computer Society Press, August 2000.

[11] S. Glassman, "A caching relay for the World Wide Web," in *Proceedings of the First International World Wide Web Conference*, 1994, pp. 69–76.

[12] C. A. Cunha, A. Bestavros, and M. E. Crovella, "Characteristics of WWW client-based traces," Boston University Department of Computer Science, Tech. Rep. TR-95-010, April 1995. [Online]. Available: http://cs-www.bu.edu/techreports/95-010-www-client-traces.ps.Z

[13] V. Almeida, A. Bestavros, M. Crovella, and A. de Oliveira, "Characterizing reference locality in the WWW," in *Proceedings of the Fourth International Conference on Parallel and Distributed Information Systems (PDIS96)*, Dec. 1996.

[14] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web Caching and Zipf-like Distributions: Evidence and Implications," in *Proceedings of IEEE Infocom*, April 1999.

[15] P. Barford and M. Crovella, "Generating representative web workloads for network and server performance evaluation," in *Proceedings of the ACM SIGMETRICS Conference*, June 1998, pp. 151–160.

[16] R. L. Mattson, J. Gecsei, D. R. Slutz, and I. L. Traiger, "Evaluation techniques for storage hierarchies," *IBM Systems Journal*, vol. 9, no. 2, pp. 78–117, 1970.

[17] L. Cherkasova and G. Ciardo, "Characterizing temporal locality and its impact on web server performance," in *Proceedings of IEEE ICCCN*, Oct. 2000, pp. 434–441.

[18] A. Balamash and M. Krunz, "Application of multifractals in the characterization of WWW traffic," in *Proceedings of the ICC 2002 Conference – Symposium on High-Speed Networks*, April 2002.

[19] D. Weikle, S. McKee, and W. Wulf, "Caches as filters: A new approach to cache analysis," in *6th Intl. Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS'98)*, July 1998.

[20] C. Williamson, "On filter effects in web caching hierarchies," *ACM Transactions on Internet Technology*, vol. 2, no. 1, pp. 47–77, Feb. 2002.

[21] R. Doyle, J. Chase, S. Gadde, and A. Vahdat, "The trickle-down effect: Web caching and server request distribution," in *Proceedings of the 6th Web Caching Workshop*, June 2001, pp. 1–18.

[22] A. Bestavros, R. L. Carter, M. E. Crovella, C. R. Cunha, A. Heddaya, and S. A. Mirdad, "Application-level document caching in the Internet," in *Proceedings of IEEE SDNE '95)*, 1995. [Online]. Available: citeseer.nj.nec.com/bestavros95applicationlevel.html

[23] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423 and 623–656, July and October 1948. [Online]. Available: "url–http://cm.bell-labs.com/cm/ms/what/shannonday/shannon1948.pdf"

[24] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. John Wiley and Sons, 1991.

[25] T. Kelly, "Thin-client web access patterns: Measurements from a cache-busting proxy," in *Proceedings of the 6th Web Caching Workshop*, Boston, MA, 2002.

[26] G. Pandurangan and E. Upfal, "Can entropy characterize performance of online algorithms?" in *Symposium on Discrete Algorithms*, 2001, pp. 727–734. [Online]. Available: citeseer.nj.nec.com/467759.html

[27] "The internet traffic archive," http://ita.ee.lbl.gov/.

[28] "W3c web characterization repository," http://repository.cs.vt.edu/.

[29] R. Fonseca, V. Almeida, M. Crovella, and B. Abrahao, "On the intrinsic locality properties of web reference streams," Boston University Computer Science Department, Tech. Rep. TR 2002-022, July 2002.