Optimizing LRU Caching for Variable Document Sizes

Predrag R. Jelenković and Ana Radovanović
Department of Electrical Engineering
Columbia University
New York, NY 10027, USA
{predrag, anar}@ee.columbia.edu
tel.: (1 212) 854 8174, 939 7157
fax: + (1 212) 932 9421

December 2002; revised December 2003

Abstract

We analyze a class of randomized Least-Recently-Used (LRU) cache replacement algorithms under the independent reference model with generalized Zipf's law request probabilities. The randomization was recently proposed for Web caching as a mechanism that discriminates between different document sizes. In particular, the cache maintains an ordered list of documents in the following way. When a document of size s is requested and found in the cache, then with probability p_s it is moved to the front of the cache; otherwise the cache stays unchanged. Similarly, if the requested document of size s is not found in the cache, the algorithm places it with probability p_s to the front of the cache or leaves the cache unchanged with the complementary probability $(1 - p_s)$. The successive randomized decisions are independent and the corresponding success probabilities p_s are completely determined by the size of the currently requested document. In the case of a replacement, the necessary number of documents that are least recently moved to the front of the cache are removed in order to accommodate the newly placed document.

In this framework, we provide explicit asymptotic characterization of the cache fault probability. Using the derived result we prove that the asymptotic performance of this class of algorithms is optimized when the randomization probabilities are chosen inversely proportional to document sizes. In addition, for this optimized and easy to implement policy, we show that its performance is within a constant factor from the optimal static algorithm.

Keywords: randomized least-recently-used caching, Zipf's law distribution, variable document sizes, Web caching, cache fault probability, averages-case analysis

This work is supported by the NSF Grant No. 0092113.
Combinatorics, Probability & Computing, to appear.

1 Introduction

Caching is widely recognized as an effective method for improving the efficiency and scalability of multimedia content delivery. It is essentially a process of keeping a smaller collection of frequently requested documents at points in the network where they can be accessed with high speed/low latency. One of the main differences between Web (World Wide Web) caching and traditional caching in computer Central Processor Units (CPUs) is that Web documents vary considerably in size depending on the type of information they carry. One of the important aspects of engineering efficient Web caching systems is designing document/file replacement algorithms that achieve high hit ratios and are easy to implement.

The cache replacement problem consists of selecting and possibly dynamically updating a collection of frequently accessed documents. The most popular replacement algorithms in practice are based on the Least-Recently-Used (LRU) cache replacement rule that possesses many desirable characteristics, including: the high hit ratio, low complexity, and flexibility to dynamically adapt to possible changes in request patterns. However, the main disadvantage of the LRU replacement algorithm is its indifference to the variability of document sizes. In order to alleviate this problem, a straightforward, randomized extension of the LRU algorithm was proposed in [11]. In this scheme, if a requested document is not found in the cache, it is either brought into the cache with probability that depends on its size or the cache content is left unchanged. In the former case, the new page is accommodated by removing the minimum necessary number of the least recently moved documents, as described in the abstract.

In this paper, assuming the independent reference model with generalized Zipf's law request probabilities, we provide an explicit asymptotic characterization of the cache fault probability. Using our asymptotic analysis and Hölder's inequality we show that the performance of this class of randomized LRU algorithms is optimized when the randomization probabilities are selected to be inversely proportional to document sizes. This algorithm, termed LRU-S, was considered in [11], but its optimality among the randomized LRU policies was not known. Furthermore, we show that LRU-S is within a constant factor from the optimal static arrangement.

Our analytical approach uses probabilistic (average-case) analysis that relies on the well-known connection between the LRU fault probability and Move-To-Front (MTF) search cost distribution, e.g. see [5, 7]. The analysis exploits a novel large deviation approach and asymptotic results that were recently developed in [7, 8]. For additional references on average case analysis of MTF and LRU algorithms see [5, 4, 7, 8]. In contrast to our probabilistic method, the majority of literature on cache replacement rules for variable document sizes is based on combinatorial (amortized, competitive) techniques. In this context, [2] represents one of the first formal treatments of the caching problem with variable-size documents; this paper proposed a new k-competitive GreedyDual-Size algorithm. For very recent results and references on competitive analysis of deterministic as well as some randomized online algorithms the reader is referred to [12, 6].

The rest of the paper is organized as follows. In Section 2, we formally introduce the MTF searching algorithm and, using the Poisson embedding technique from [4], derive a representation result for the MTF search cost distribution. Then, in Theorem 2 of Section 3, we present our main asymptotic result that characterizes the MTF search cost distribution for the generalized Zipf's law requests. Using this result, we show in Theorem 3 of Section 4 that LRU-S algorithm has asymptotically the best performance within the class of randomized LRU rules. Furthermore, in the same section, we prove that LRU-S policy is within a constant factor from the optimal static arrangement. The concluding remarks are presented in Section 5.

2 Randomized move-to-front algorithm

From a mathematical perspective the analyses of the LRU caching and MTF searching algorithms are essentially the same; this will be formally explained in Section 4. Thus, we proceed with defining a randomized MTF algorithm in this section. Consider a finite list of N documents with M possible sizes, $s_1, s_2, \dots, s_M$ that is dynamically updated as follows. When a document of size $s_k, 1 \le k \le M$ is requested, it is either moved to the front of the list with a positive probability p_k or the list is left unchanged with a complementary probability $1 - p_k$. In the former case, documents that were in front of the requested item are moved down the list to provide the necessary space at the top of the list. These randomized decisions are independent and their success probabilities p_k are determined by the size of the currently accessed document.

Assume that the list of N documents (items) is enumerated as $L = \{(i,k), 1 \le i \le N_k, 1 \le k \le M\}$, where N_k is the total number of documents of size s_k, $N_1 + N_2 + \dots + N_M = N$ and $\mathbf{N} = (N_1, \dots, N_M)$. Request process $\{(R_n, I_n)\}_{n=-\infty}^{\infty}$ is a sequence of i.i.d. random variables, where an event $\{R_n = i, I_n = k\}$ represents a request for document (i,k) having size s_k at time n. We denote request probabilities as $\mathbb{P}[R_n = i, I_n = k] = q_i^{(k)}$ and, without loss of generality, assume $q_1^{(k)} \ge q_2^{(k)} \ge \dots$ for all $1 \le k \le M$. The performance quantity of interest for this algorithm is the search cost $C_n^{\mathbf{N}}$ that represents the total size of all documents in the list that are in front of the requested item. On event $\{R_n = i, I_n = k\}$, $C_n^{\mathbf{N}}$ is equal to the total size of documents in the list that are in front of document (i,k). Our objective in this paper is to characterize the distribution of the stationary search cost $C^{\mathbf{N}}$ and use this result to estimate the cache fault probability of the corresponding randomized LRU replacement scheme. Clearly, the stationary search cost $C^{\mathbf{N}}$ exists since the process of permutations on the elements of the list represents an irreducible and aperiodic finite state Markov chain. Note that this existence can be poved in a more general setting of stationary and ergodic requests using the same arguments as in the proof of Lemma 1 of [8]. Throughout the paper we assume that the search cost process is in stationarity.

In order to facilitate the analysis, we use the Poisson embedding technique that was first introduced for the MTF searching problem in [4]. The stationary process $\{(R_n, I_n), C_n^{\mathbf{N}}\}_{n=-\infty}^{\infty}$ is embedded into an independent sequence of Poisson points $\{\tau_n, -\infty < n < \infty\}$ of unit rate. We observe the search cost at time τ_0 that, without loss of generality, is set to $\tau_0 = 0$. Then, due to the Poisson decomposition theorem, the request process for document (i,k) is Poisson with rate $q_i^{(k)}$. Similarly, if $N_i^{(k)}(u,t)$ denotes the number of requests for document (i,k) in (u,t) that resulted in moving (i,k) to the front of the list, the Poisson decomposition theorem implies that this process is Poisson with rate $q_i^{(k)} p_k$. Next, we define $B_j^{(k)}(t) = 1[N_j^{(k)}(-t,0) > 0]$, $t > 0$ with $\mathbb{P}[B_i^{(k)}(t) = 1] = 1 - e^{-q_i^{(k)} p_k t}$, which indicates whether document (i,k) is moved at least once to the front of the list in $(-t,0)$. Therefore, the total size of different documents, not equal to (i,k), that are moved to the front of the list in time $(-t,0)$ is equal to

$$S_i^{(k)}(t; \mathbf{N}) = \sum_{(j,l) \ne (i,k)} s_l B_j^{(l)}(t). \tag{1}$$

Note that $S_i^{(k)}(t; \mathbf{N})$ is monotonic in t, since it is a nondecreasing function of the counting processes $N_j^{(l)}(-t,0)$.

Next, let $-T(i,k)$ be the last time before $\tau_0 = 0$ when item (i,k) was moved to the front of the list and note that its distribution is equal to

$$\mathbb{P}[T(i,k) > t] = \mathbb{P}[N_i^{(k)}(-t,0) = 0] = e^{-q_i^{(k)} p_k t}. \tag{2}$$

Note that for each (i,k) the process $\{S_i^{(k)}(t; N)\}_{t>0}$ is independent of $T(i,k)$, and all of the processes $S_i^{(k)}$ and $T(i,k)$ are independent of (R_0, I_0). Thus, given that at time $\tau_0 = 0$ we request an item $(R_0, I_0) = (i,k)$, it follows from (1) that the conditional search cost is equal to $S_i^{(k)}(T(i,k);N)$. Hence, we have derived the following representation result.

Theorem 1 *The stationary search cost satisfies*

$$C^{\mathbf{N}} \stackrel{d}{=} S_{R_0}^{(I_0)}(T;\mathbf{N}), \tag{3}$$

where $\stackrel{d}{=}$ *denotes equality in distribution and* $T \equiv T(R_0, I_0)$.

Remark: For the case of the ordinary MTF algorithm (same document sizes), this result was first derived using formal languages in [5] and later reproduced in [4]. $\diamond$

Next, for a fixed number of possible document sizes M, we investigate the behavior of the stationary search cost $C^{\mathbf{N}}$ as $\mathbf{N} \to \infty$, where $\mathbf{N} \to \infty$ means $\min_k N_k \to \infty$; recall that N_k represents the total number of documents of size s_k. In this regard, consider an infinite sequence of probabilities $q_i^{(k)}$, $i \geq 1$, $1 \leq k \leq M$, such that $\sum_{k=1}^{M}\sum_{i=1}^{\infty} q_i^{(k)} = 1$ and, for all k, $q_i^{(k)}$is nonincreasing in i. Now, we define a sequence of request processes $\{(R_n^{\mathbf{N}}, I_n^{\mathbf{N}})\}$, with

$$q_{i,\mathbf{N}}^{(k)} = \frac{q_i^{(k)}}{\sum_{l=1}^{M}\sum_{i=1}^{N_l} q_i^{(l)}}, \quad 1 \leq i \leq N_k, 1 \leq k \leq M. \tag{4}$$

Let $C^{\mathbf{N}}$ be the corresponding stationary search cost defined by request probabilities $q_{i,\mathbf{N}}^{(k)}$.

Furthermore, consider an infinite list of items accessed by the limiting request process $(R_n^{\infty}, I_n^{\infty}) \equiv (R_n, I_n)$ with probabilities $q_i^{(k)}$. Then, similarly as in the finite list case, let $B_i^{(k)}(t)$ be a Bernoulli random variable indicating the event that item (i,k) is moved to the front of the list at least once in $(-t, 0)$, and define $S_i^{(k)}(t) = \sum_{(j,l)\neq(i,k)} s_l B_j^{(l)}(t)$. The variable $-T(i,k)$ represents the time of the last move of document (i,k) to the front of the list with its distribution given by (2). Note that these variables are well defined and almost surely finite. Now, similarly as in Proposition 4.4 of [3] for the case of i.i.d. requests for documents of the same size, using dominated convergence one can easily prove the following limit.

Proposition 1 *The constructed sequence of stationary search costs* $C^{\mathbf{N}}$ *converges in distribution to* C, *i.e. as* $\mathbf{N} \to \infty$

$$C^{\mathbf{N}} \stackrel{d}{\Rightarrow} C \triangleq S_{R_0}^{(I_0)}(T), \tag{5}$$

where $T \equiv T(R_0, I_0)$.

The following section characterizes the asymptotic behavior of the tail of the searching cost C.

In this paper, using the standard notation, for any two real functions $a(t)$ and $b(t)$ and fixed $t_0 \in \mathbb{R} \cup \{\infty\}$ the expression $a(t) \sim b(t)$ as $t \to t_0$ denotes $\lim_{t\to t_0} a(t)/b(t) = 1$. Similarly, we say that $a(t) \gtrsim b(t)$ as $t \to t_0$, if $\liminf_{t\to t_0} a(t)/b(t) \geq 1$; $a(t) \lesssim b(t)$ has a complementary definition. In addition, throughout the paper H denotes a sufficiently large positive constant. The values of H may differ in different places, for example, $H/2 = H$, $H^2 = H$, $H + 1 = H$, etc.

3 Main results

Our results are derived under the following assumption of generalized Zipf's law distribution. This distribution has been widely used for modeling cache request patterns and, in particular, for capturing the Web access behavior, e.g. see [1, 9].

Assumption 1 *For any fixed* $1 \le m \le M$, *let* $q_i^{(k)} \sim c_k/i^\alpha, c_k > 0$, $1 \le k \le m$ *and* $q_i^{(k)} = o(1/i^\alpha)$, $m < k \le M$ *as* $i \to \infty$, $\alpha > 1$.

Theorem 2 *Under Assumption 1,*

$$\lim_{x\to\infty} \mathbb{P}[C > x]x^{\alpha-1} = \frac{K(\alpha)}{(\alpha-1)}\left(c_1^{1/\alpha}p_1^{1/\alpha}s_1 + \cdots + c_m^{1/\alpha}p_m^{1/\alpha}s_m\right)^{\alpha-1}\left(\frac{c_1^{1/\alpha}}{p_1^{1-1/\alpha}} + \cdots + \frac{c_m^{1/\alpha}}{p_m^{1-1/\alpha}}\right), \quad (6)$$

where

$$K(\alpha) \triangleq \left(1 - \frac{1}{\alpha}\right)\left[\Gamma\left(1 - \frac{1}{\alpha}\right)\right]^\alpha. \quad (7)$$

Remarks: (i) The constant $K(\alpha)$ is monotonically increasing in α with $\lim_{\alpha\to\infty} K(\alpha) = e^\gamma \approx 1.78$ and $\lim_{\alpha\to 1} K(\alpha) = 1$, where γ is the Euler constant; this was formally proved in Theorem 3 of [7]. (ii) Clearly, by setting $p_k \equiv 1, s_k \equiv 1$, the theorem reduces to the traditionally studied MTF lists (LRU system), investigated in Theorem 3 of [7]. Note that by enumerating the set of probabilities $\{q_i^{(k)}\}$ in their nonincreasing order, $q'_1 \ge q'_2 \ge \ldots$, easy, but somewhat tedious, algebra shows that the tail of the resulting request distribution satisfies $\sum_{i\ge x} q'_i \sim (c_1^{1/\alpha} + \cdots + c_m^{1/\alpha})^\alpha/((\alpha-1)x^{\alpha-1})$ as $x \to \infty$. $\diamond$

In order to prove the preceding result we use Lemmas 1 and 2 of [7] and Lemma 4 of [8]; for reasons of completeness, we state these lemmas in the appendix.

Proof: Equation (5) easily implies

$$\begin{aligned}\mathbb{P}[C > x] &= \mathbb{P}[S_{R_0}^{(I_0)}(T(R_0, I_0)) > x] \\ &= \sum_{i=1}^{\infty}\sum_{k=1}^{M} \mathbb{P}[S_i^{(k)}(T(i,k)) > x, R_0 = i, I_0 = k] \\ &= \int_0^\infty \sum_{i=1}^{\infty}\sum_{k=1}^{M} (q_i^{(k)})^2 p_k e^{-q_i^{(k)}p_k t}\mathbb{P}[S_i^{(k)}(t) > x]\, dt, \end{aligned} \quad (8)$$

where the last equality follows from the independence of $T(i,k)$, $\{S_i^{(k)}(t)\}_{t>0}$ and (R_0, I_0), and expression (2). Let $S^{(l)}(t) = \sum_{i=1}^\infty B_i^{(l)}(t)$ and $S(t) = \sum_{l=1}^M s_l S^{(l)}(t)$, representing the total size of distinct documents that are moved to the front of the list in $(-t, 0)$. Then, $S(t) \le S_i^{(k)}(t) + s_k \le S_i^{(k)}(t) + s$, where $s \triangleq \max_k s_k$.

We proceed with proving the *lower bound* first. From (8) we obtain

$$\mathbb{P}[C > x] \ge \int_{g_{\epsilon_+}x^\alpha}^\infty \sum_{k=1}^{m}\sum_{i=1}^{\infty} (q_i^{(k)})^2 p_k e^{-q_i^{(k)}p_k t}\mathbb{P}[S(t) > x + s]\, dt,$$

where g_{ϵ_+} is chosen according to

$$g_{\epsilon_+} \triangleq \frac{(1+2\epsilon)^\alpha}{(\Gamma\left[1 - \frac{1}{\alpha}\right])^\alpha a^\alpha}$$

with

$$a \triangleq s_1 c_1^{1/\alpha}p_1^{1/\alpha} + \cdots + s_m c_m^{1/\alpha}p_m^{1/\alpha}. \quad (9)$$

The preceding inequality and the monotonicity of $S(t)$ (see the comment after equation (1)) yield

$$\mathbb{P}[C > x] \geq \mathbb{P}[S(g_{\epsilon_+} x^\alpha) > x + s] \int_{g_{\epsilon_+} x^\alpha}^{\infty} \sum_{k=1}^{m} \sum_{i=1}^{\infty} (q_i^{(k)})^2 p_k e^{-q_i^{(k)} p_k t} \, dt. \tag{10}$$

From Assumption 1 and Lemma 5 of the appendix, it is easy to show that

$$\mathbb{E} \sum_{k=1}^{m} s_k S^{(k)}(g_{\epsilon_+} x^\alpha) \sim (1 + 2\epsilon) x \;\text{ as }\; x \to \infty. \tag{11}$$

Furthermore, by Assumption 1, for any $\epsilon > 0$, there exists i_ϵ such that for all $i \geq i_\epsilon$, the request probabilities are bounded as $q_i^{(k)} \leq \frac{\epsilon}{i^\alpha}$, $m < k \leq M$, and, therefore, Lemma 5 of the appendix yields

$$\begin{aligned}\limsup_{t\to\infty} t^{-1/\alpha} \mathbb{E} S^{(k)}(t) &\leq \limsup_{t\to\infty} t^{-1/\alpha} \sum_{i=1}^{i_\epsilon} (1 - e^{-q_i^{(k)} p_k t}) + \limsup_{t\to\infty} t^{-1/\alpha} \sum_{i=i_\epsilon+1}^{\infty} (1 - e^{-t p_k \epsilon / i^\alpha}) \\ &\leq H \epsilon^{1/\alpha}. \end{aligned} \tag{12}$$

Thus, since ϵ can be chosen arbitrarily small, we obtain for $m < k \leq M$,

$$\mathbb{E} S^{(k)}(t) = o(t^{\frac{1}{\alpha}}) \;\text{ as }\; t \to \infty. \tag{13}$$

Then, the estimates in (11) and (13) yield $\mathbb{E} S(g_{\epsilon_+} x^\alpha) = \sum_{k=1}^{M} s_k \mathbb{E} S^{(k)}(g_{\epsilon_+} x^\alpha) \geq (1 + \epsilon)(x + s)$ for x large enough ($x \geq x_\epsilon$). Hence, by Lemma 6 of the appendix, for any $\epsilon > 0$ and x large enough

$$\mathbb{P}[S(g_{\epsilon_+} x^\alpha) \geq x + s] > 1 - \epsilon.$$

By replacing the last inequality in (10), we obtain for x large enough

$$\mathbb{P}[C > x] \geq (1 - \epsilon) \sum_{k=1}^{m} \frac{1}{p_k} \int_{g_{\epsilon_+} x^\alpha}^{\infty} \sum_{i=1}^{\infty} (q_i^{(k)} p_k)^2 e^{-q_i^{(k)} p_k t} \, dt. \tag{14}$$

Now, by exploiting Lemma 4 of the appendix and Assumption 1, we infer that for t large enough ($t \geq t_\epsilon$) and all $1 \leq k \leq m$

$$\sum_{i=1}^{\infty} (q_i^{(k)} p_k)^2 e^{-q_i^{(k)} p_k t} \geq (1 - \epsilon) \frac{(p_k c_k)^{1/\alpha}}{\alpha} \Gamma\left[2 - \frac{1}{\alpha}\right] t^{-2 + \frac{1}{\alpha}}. \tag{15}$$

Now, by replacing (15) into (14) we obtain

$$\mathbb{P}[C > x] \geq (1 - \epsilon)^2 \sum_{k=1}^{m} \frac{1}{p_k} \int_{g_{\epsilon_+} x^\alpha}^{\infty} \frac{(p_k c_k)^{1/\alpha}}{\alpha} \Gamma\left[2 - \frac{1}{\alpha}\right] t^{-2 + 1/\alpha} \, dt.$$

Hence, after computing the integral in the preceding expression, multiplying it with $x^{\alpha - 1}$, taking the limit as $x \to \infty$, and passing $\epsilon \downarrow 0$, we conclude

$$\liminf_{x\to\infty} \mathbb{P}[C > x] x^{\alpha - 1} \geq \frac{K(\alpha)}{(\alpha - 1)} \left(c_1^{1/\alpha} p_1^{1/\alpha} s_1 + \cdots + c_m^{1/\alpha} p_m^{1/\alpha} s_m \right)^{\alpha - 1} \left(\frac{c_1^{1/\alpha}}{p_1^{1 - 1/\alpha}} + \cdots + \frac{c_m^{1/\alpha}}{p_m^{1 - 1/\alpha}} \right). \tag{16}$$

Next, we proceed with the proof of the *upper bound*. Since $S_i^{(k)}(t) \le S(t)$, after splitting the integral in (8), we obtain

$$\begin{aligned}
\mathbb{P}[C > x] &\le \sum_{k=1}^{M} \int_0^\infty \sum_{i=1}^{\infty} (q_i^{(k)})^2 p_k e^{-q_i^{(k)} p_k t} \mathbb{P}[S(t) > x]\, dt \\
&\le \sum_{k=1}^{M} \frac{1}{p_k} \left(\int_0^{g_{\epsilon_} x^\alpha} \sum_{i=1}^{\infty} (q_i^{(k)} p_k)^2 e^{-q_i^{(k)} p_k t} \mathbb{P}[S(t) > x]\, dt + \int_{g_{\epsilon_} x^\alpha}^{\infty} \sum_{i=1}^{\infty} (q_i^{(k)} p_k)^2 e^{-q_i^{(k)} p_k t}\, dt \right) \\
&\triangleq \sum_{k=1}^{M} \frac{1}{p_k} (I_1^{(k)}(x) + I_2^{(k)}(x)), \qquad (17)
\end{aligned}$$

where $g_{\epsilon\_}$ is defined as

$$g_{\epsilon\_} \triangleq \frac{(1-2\epsilon)^\alpha}{(\Gamma\left[1 - \frac{1}{\alpha}\right])^\alpha a^\alpha}$$

with a being the same as in (9). Then, by using similar arguments as in (11) and (13), we conclude that $\mathbb{E}S(g_{\epsilon\_} x^\alpha) \le (1-\epsilon)x$ for all x large enough. Therefore, by Lemma 6 of the appendix, for all x large enough ($x \ge x_\epsilon$)

$$\mathbb{P}[S(g_{\epsilon\_} x^\alpha) > x] \le H e^{-h\theta_\epsilon x}. \qquad (18)$$

Thus, since $S(t)$ is nondecreasing and $\int_0^\infty \sum_{i=1}^\infty (q_i^{(k)} p_k)^2 e^{-q_i^{(k)} p_k t}\, dt = \sum_{i=1}^\infty q_i^{(k)} p_k \le 1$, we conclude that for any $1 \le k \le M$

$$\begin{aligned}
I_1^{(k)}(x) &\le \mathbb{P}[S(g_{\epsilon_} x^\alpha) > x] g_{\epsilon_} x^\alpha \\
&= o(x^{-\alpha+1}) \ \text{ as } \ x \to \infty, \qquad (19)
\end{aligned}$$

where the last equality follows from (18).

Now, from Assumption 1 for all $m < k \le M$, any $\epsilon > 0$ and i large enough (say $i \ge i_\epsilon$) we obtain $q_i^{(k)} p_k \le \epsilon / i^\alpha$. Thus, for x large enough and $k > m$, we estimate the integral

$$\begin{aligned}
I_2^{(k)}(x) &\le \sum_{i=1}^{\lfloor \epsilon^{1/\alpha} x \rfloor} q_i^{(k)} p_k e^{-q_i^{(k)} p_k g_{\epsilon_} x^\alpha} + \sum_{i=\lfloor \epsilon^{1/\alpha} x \rfloor + 1}^{\infty} \frac{\epsilon}{i^\alpha} \\
&\le \frac{1}{g_{\epsilon_} x^\alpha} \sum_{i=1}^{\lfloor \epsilon^{1/\alpha} x \rfloor} q_i^{(k)} p_k g_{\epsilon_} x^\alpha e^{-q_i^{(k)} p_k g_{\epsilon_} x^\alpha} + \int_{\epsilon^{1/\alpha} x}^{\infty} \frac{\epsilon}{u^\alpha}\, du \\
&\le \frac{\epsilon^{1/\alpha}}{x^{\alpha-1}} \left[\frac{e^{-1}}{g_{\epsilon_}} + \frac{1}{\alpha - 1} \right],
\end{aligned}$$

where in the last inequality we exploited the fact that $\sup_{y \ge 0} y e^{-y} = e^{-1}$. Since ϵ can be arbitrarily small, we infer that for all $m < k \le M$

$$I_2^{(k)}(x) = o(x^{-\alpha+1}) \ \text{ as } \ x \to \infty. \qquad (20)$$

Next, similarly as in (15), for $1 \le k \le m$ and x large we derive

$$I_2^{(k)}(x) \le (1+\epsilon) \int_{g_{\epsilon\_} x^\alpha}^{\infty} \frac{(p_k c_k)^{1/\alpha}}{\alpha} \Gamma\left[2 - \frac{1}{\alpha}\right] t^{-2+1/\alpha}\, dt. \qquad (21)$$

Now, replacing the estimates (19), (20) and (21) in (17) and using analogous steps to those taken in estimating (16) yield the upper bound

$$\limsup_{x\to\infty} \mathbb{P}[C > x]x^{\alpha-1} \leq \frac{K(\alpha)}{(\alpha-1)} \left(c_1^{1/\alpha}p_1^{1/\alpha}s_1 + \cdots + c_m^{1/\alpha}p_m^{1/\alpha}s_m\right)^{\alpha-1} \left(\frac{c_1^{1/\alpha}}{p_1^{1-1/\alpha}} + \cdots + \frac{c_m^{1/\alpha}}{p_m^{1-1/\alpha}}\right).$$

Finally, the last inequality and (16) prove the theorem. $\diamond$

4 The optimal randomized LRU algorithm

Consider a universe of a large number of documents N, a subset of which can be placed in an easily accessible location of size x, called cache. Interested parties request these documents by searching the cache first and then, if a document is not found there, the outside universe. When a document is not found in the cache, we say that there was a *cache fault* or *miss*. In addition, at the time of a miss, a cache replacement algorithm may replace the necessary number of documents in the cache with a newly retrieved document. The replacements are done in order to reduce the number of future faults. This system models the Web caching environment where the parties that request documents correspond to end-users, the cache represents a local proxy server and the document universe is the entire World Wide Web. The minor difference between this caching system and the traditional computer CPU caching one is that cache updates in the case of faults are only optional. In other words, when a cache fault occurs, the requested document can be retrieved directly from its original storage (server) without impacting the cache content. Furthermore, we consider the so-called on-line algorithms that make their replacement decisions based only on the knowledge of the past requests.

We assume that the request process is the same as in the case of the randomized MTF algorithm described in Section 2. Similarly, the decisions of the corresponding randomized LRU cache replacement algorithm depend on the currently requested document size $s_k, 1 \leq k \leq M$. More specifically, the cache maintains an ordered list of items in the following way. When a document of size s_k, $1 \leq k \leq M$, is requested and found in the cache (cache hit), then with probability p_k it is moved to the front of the cache; otherwise the cache stays unchanged. In the case of a cache miss for an item of size s_k, the algorithm places the requested item with probability p_k to the front of the cache or leaves the cache unchanged with the complementary probability $(1-p_k)$. The successive randomized decisions are independent and the success probabilities p_k are completely determined by the size of the currently requested document s_k. In the case of a replacement, the necessary number of documents that are least recently moved to the front of the cache are removed in order to accommodate the newly placed document.

We investigate the behavior of the randomized LRU cache fault probability $P^{\mathbf{N}}(x)$ using the connection between the randomized LRU caching and MTF searching algorithms. Since the fault probability does not depend on the arrangement of documents that are outside of the cache, they can be arranged in an increasing order of the last times they were moved to the front of the cache. Thus, the ordered list of documents inside and outside of the cache under the randomized LRU rule behaves the same as the corresponding randomized MTF scheme. Clearly, using the notation from Section 2, the stationary cache fault probability satisfies

$$\mathbb{P}[C^{\mathbf{N}} > x] \leq P^{\mathbf{N}}(x) = \mathbb{P}[S_{R_0}^{(I_0)}(T;\mathbf{N}) + s_{I_0} > x] \leq \mathbb{P}[C^{\mathbf{N}} > x - s],$$

where $s = \max_k s_k$. Now, similarly as in (4), we can construct a family of caching systems indexed by $\mathbf{N}$ and show that the limiting cache fault probability $P(x) = \lim_{\mathbf{N}\to\infty} P^{\mathbf{N}}(x)$ exists, since $S_{R_0}^{(I_0)}(T;\mathbf{N}) \to$

$S_{R_0}^{(I_0)}(T)$ as $N \to \infty$. Furthermore, this probability satisfies $\mathbb{P}[C > x] \leq P(x) \leq \mathbb{P}[C > x - s]$, which, in conjunction with Theorem 2, implies

Lemma 1 *Under Assumption 1*

$$P(x) \sim \mathbb{P}[C > x] \ \text{ as } \ x \to \infty.$$

Remark: Note that, even without any analogy with MTF lists, it can be seen that event $\{S_{R_0}^{(I_0)}(T)+s_{I_0} > x\}$ represents a cache fault. In the caching context, $S_{R_0}^{(I_0)}(T)$ represents the total size of all different documents that are placed in the first position of the cache since the last time the currently requested document was at the front of the cache. Therefore, if this variable plus the size of a currently requested document s_{I_0} is larger than x, document (R_0, I_0) had to have been evicted from the cache at some point in the past. However, the analogy with lists is useful since it ties these two different applications of caching and searching together. $\diamond$

The following theorem shows that the optimal performance of the randomized LRU replacement scheme is achieved when randomization probabilities are inversely proportional to document sizes. Let $P_s(x)$ be the fault probability for this particular selection of randomization probabilities. Adopting the notation from [11], we will refer to this policy as LRU-S.

Theorem 3 *Under Assumption 1, asymptotically optimal randomized LRU replacement scheme that minimizes the expression in (6) is achieved when $p_k = 1/s_k$, $1 \leq k \leq M$. Furthermore, for this choice of randomization probabilities, the fault probability satisfies*

$$\lim_{x \to \infty} x^{-\alpha+1} P_s(x) = \frac{K(\alpha)}{(\alpha - 1)} \left(c_1^{\frac{1}{\alpha}} s_1^{1-\frac{1}{\alpha}} + \cdots + c_m^{\frac{1}{\alpha}} s_m^{1-\frac{1}{\alpha}} \right)^{\alpha}. \tag{22}$$

Remarks: (i) Note that this algorithm can provide an arbitrarily large relative improvement in comparison to the ordinary LRU scheme. In this regard, by setting $p_k \equiv 1$ in (6), one obtains the asymptotic result for the traditional LRU. Then, computing the ratio between the obtained constant in (6) and (22), setting $c_k \equiv c$ and letting (say) $s_m \to \infty$, one easily calculates this limit to be equal to m, which can be made arbitrarily large. (ii) Note that the implementation of the LRU-S algorithm does not require the knowledge of the probabilities $\{q_i^{(k)}\}$. The only information needed is the size of the currently requested document, which is available in most applications. However, the optimality of the LRU-S algorithm beyond Assumption 1 remains an open problem. $\diamond$

Proof: Minimizing the expression in (6) is equivalent to optimizing the following function

$$f(p) \triangleq \left(c_1^{1/\alpha} p_1^{1/\alpha} s_1 + \cdots + c_m^{1/\alpha} p_m^{1/\alpha} s_m \right)^{\alpha-1} \left(\frac{c_1^{1/\alpha}}{p_1^{1-1/\alpha}} + \cdots + \frac{c_m^{1/\alpha}}{p_m^{1-1/\alpha}} \right).$$

Next, define $a_i \triangleq (c_i^{1/\alpha}/p_i^{1-1/\alpha})^{1/\alpha}$ and $b_i \triangleq (c_i^{1/\alpha} p_i^{1/\alpha} s_i)^{1-1/\alpha}$, $1 \leq i \leq M$. Then, Hölder's inequality implies

$$\begin{aligned} f(p)^{1/\alpha} &= (a_1^{\alpha} + \cdots + a_m^{\alpha})^{\frac{1}{\alpha}} (b_1^{\frac{\alpha}{\alpha-1}} + \cdots + b_m^{\frac{\alpha}{\alpha-1}})^{1-\frac{1}{\alpha}} \\ &\geq a_1 b_1 + \cdots + a_m b_m \\ &= c_1^{\frac{1}{\alpha}} s_1^{1-\frac{1}{\alpha}} + \cdots + c_m^{\frac{1}{\alpha}} s_m^{1-\frac{1}{\alpha}}, \end{aligned} \tag{23}$$

where the equality is achieved for $a_i^{\alpha-1} = b_i$, i.e. $p_i = 1/s_i$, $1 \le i \le m$. For $m < i \le M$, since the performance of the algorithm does not depend on p_i, one can choose arbitrary randomization probabilities p_i, $i > m$. However, for reasons of consistency, we select $p_i = 1/s_i$ as well. Furthermore, (23) and (6) yield (22). $\diamond$

4.1 Asymptotically optimal greedy static algorithm

Consider a static algorithm that maximizes the probability of finding a document in the cache. In other words, this algorithm places a collection of documents $\mathcal{C}$ in the cache such that the probability $\sum_{(i,k)\in\mathcal{C}} q_i^{(k)}$ is maximized under the constraint that $\sum_{(i,k)\in\mathcal{C}} s_k \le x$. After this selection is made the content of the cache is never changed. We term this algorithm *optimal static* (OS) and denote its fault probability as $P_o(x)$. It is intuitively straightforward that, in the context of the independent reference model, this algorithm is optimal (minimizes the cache fault probability) among all on-line caching schemes, as described previously at the beginning of Section 4. To see this more formally, consider a class of on-line idealized prefetching (IP) algorithms where after every request one is allowed to change the content of the whole cache at no extra cost. Let $P_{op}(x)$ and $P_{oc}(x)$ be the limiting long-term average fault probabilities of optimal IP and caching algorithms, respectively (these averages exist since we will show that $P_{oc}(x) = P_{op}(x) = P_o(x)$). Clearly, since the action space of IP algorithms is larger than the one for caching algorithms, $P_{oc}(x) \ge P_{op}(x)$. Also, since IP algorithms can replace the whole content of the cache at no extra cost, the action of an optimal IP algorithm at the time of (say) the nth request that minimizes the *long-term fault probability* is the same as the one that minimizes the *one-step fault probability* at the time of the $(n+1)$st request. Furthermore, because of the independence of the requests, it is clear that the OS algorithm minimizes this one-step fault probability and, thus, represents the optimal IP algorithm. Furthermore, since the OS algorithm belongs to the class of caching algorithms and $P_{oc}(x) \ge P_{op}(x)$, OS is also an optimal on-line caching algorithm.

Now, we introduce a *greedy static* algorithm that places documents in the cache according to a decreasing order of $q_i^{(k)}/s_k$ (see page 263 of [10]) with $P_{os}(x)$ being the corresponding fault probability for the cache of size x. Using Assumption 1, we show that the performance of this algorithm is asymptotically equal to the optimal static algorithm. Without loss of generality assume $s_1 = \min s_k = 1$. Let $a \vee b = \max(a,b)$ and $a \wedge b = \min(a,b)$.

Lemma 2 *Under Assumption 1,*

$$P_o(x) \gtrsim \frac{(c_1^{1/\alpha} + \cdots + c_m^{1/\alpha} s_m^{1-1/\alpha})^\alpha}{(\alpha-1)x^{\alpha-1}} \quad \textit{as} \quad x \to \infty.$$

Proof: Since $q_1^{(k)} \ge q_2^{(k)} \ge \dots$, $1 \le k \le M$, the fault probability of the optimal static algorithm for the cache of size x can be expressed as

$$P_o(x) = \sum_{i=N_1^*}^{\infty} q_i^{(1)} + \cdots + \sum_{i=N_M^*}^{\infty} q_i^{(M)},$$

where $N_1^* - 1, \dots, N_M^* - 1$ represent the optimal numbers of stored documents of sizes $1, \dots, s_M$, respectively, satisfying the constraint $N_1^* - 1 + \cdots + (N_M^* - 1)s_M \le x$. From the assumption of the theorem, for any $\epsilon > 0$, there exists i_ϵ, such that for all $i \ge i_\epsilon$, $q_i^{(k)} \ge (1-\epsilon)c_k/i^\alpha$, $1 \le k \le m$. Therefore, we can lower bound $P_o(x)$ as

$$
\begin{aligned}
P_o(x) &\geq \min_{N_1^*,\ldots,N_m^*} \sum_{N_1^* \vee i_\epsilon}^{\infty} q_i^{(1)} + \cdots + \sum_{N_m^* \vee i_\epsilon}^{\infty} q_i^{(m)} \\
&\geq \frac{1-\epsilon}{\alpha-1} \min_{N_1^*,\ldots,N_m^*} \sum_{k=1}^{m} \frac{c_k}{(N_k^* \vee i_\epsilon)^{\alpha-1}},
\end{aligned} \tag{24}
$$

where the minimum is subject to the constraint $N_1^* - 1 + s_2(N_2^* - 1) + \cdots + s_m(N_m^* - 1) \leq x$. The last constraint is further relaxed by $N_1^* \vee i_\epsilon + \cdots + s_m N_m^* \vee i_\epsilon \leq x + (i_\epsilon + 1)\sum_{k=1}^{M} s_k$ and, therefore, (24) is further lower bounded by

$$
P_o(x) \geq \frac{1-\epsilon}{\alpha-1} \min_{u_1,\ldots,u_m} \sum_{k=1}^{m} \frac{c_k}{u_k^{\alpha-1}},
$$

subject to $\sum_{k=1}^{m} u_k s_k \leq x + \Delta$, $\Delta = (i_\epsilon + 1)\sum_{k=1}^{m} s_k$, $u_k \triangleq N_k^* \vee i_\epsilon$. Then, by allowing u_k-s to be real valued, the minimum becomes potentially even smaller. Thus, solving the continuous optimization problem using Lagrange multipliers results in

$$
\liminf_{x\to\infty} x^{\alpha-1} P_o(x) \geq (1-\epsilon)\frac{(c_1^{1/\alpha} + \cdots + c_m^{1/\alpha} s_m^{1-1/\alpha})^\alpha}{(\alpha-1)},
$$

which, by passing $\epsilon \to 0$, proves the result. $\diamond$

In the following theorem we show that the performance of the proposed greedy static and optimal policy are asymptotically the same.

Theorem 4 *Under Assumption 1,*

$$
P_{os}(x) \sim P_o(x) \sim \frac{(c_1^{1/\alpha} + \cdots + c_m^{1/\alpha} s_m^{1-1/\alpha})^\alpha}{(\alpha-1)x^{\alpha-1}} \quad \text{as} \quad x \to \infty.
$$

The following lemma will be used in the proof of the preceding theorem.

Lemma 3 *Let $N_k(x)$ be the number of documents of size s_k, $1 \leq k \leq M$, that is placed in the cache of size x according to the greedy static rule. Assume that document classes with finite probability support, i.e. those having $q_i^{(k)} = 0$ for some i, are enumerated between $n_0 < k \leq M$. Then, for x large enough, all documents with positive probabilities of size s_k, $n_0 < k \leq M$ are placed in the cache, while for $1 \leq k \leq n_0$*

$$
\lim_{x\to\infty} N_k(x) = \infty. \tag{25}
$$

Proof: It is easy to see that for each k, $N_k(x)$ is nondecreasing in x and therefore $\lim_{x\to\infty} N_k(x)$ exists for all $1 \leq k \leq M$. Furthermore, since $\sum_k N_k(x)s_k + 1 > x$, there exists $1 \leq k^* \leq M$, such that (25) holds. Then, for any $1 \leq l \leq M$ and all x, the greedy algorithm implies

$$
0 \leq \frac{q_{N_l(x)+1}^{(l)}}{s_l} \leq \frac{q_{N_{k^*}(x)}^{(k^*)}}{s_{k^*}}.
$$

This inequality, since $\lim_{x\to\infty} q_{N_{k^*}(x)}^{(k^*)} = 0$, implies that either $q_{N_l(x)+1}^{(l)} \equiv 0$ or $q_{N_l(x)+1}^{(l)} \downarrow 0$ over positive values as $x \to \infty$. The former corresponds to classes l ($n_0 < l \leq M$) having finite probability support, while the latter is only possible if $1 \leq l \leq n_0$ and (25) holds. $\diamond$

Proof of Theorem 4: Let n_0, $N_k(x)$, $1 \leq k \leq M$, be defined as in Lemma 3. Assumption 1 implies that for any $1 > \epsilon > 0$ and all $1 \leq j \leq n_0$ there exists i_ϵ, such that for all $i \geq i_\epsilon$

$$\frac{(1-\epsilon)c_k}{i^\alpha} \leq q_i^{(k)} \leq \frac{(1+\epsilon)c_k}{i^\alpha} \;\text{ for } k = 1, \ldots, m, \tag{26}$$

and

$$q_i^{(k)} \leq \frac{\epsilon}{i^\alpha} \;\text{ for } k = m+1, \ldots, n_0. \tag{27}$$

In addition, by Lemma 3, for all x large enough $N_j(x) > i_\epsilon, 1 \leq j \leq n_0$. Next, by exploiting Lemma 3 and expressions (26) and (27) we upper bound $P_{os}(x)$ for x large enough

$$\begin{aligned}
P_{os}(x) &= \sum_{i=N_1(x)+1}^{\infty} q_i^{(1)} + \cdots + \sum_{i=N_{n_0}(x)+1}^{\infty} q_i^{(n_0)} \\
&\leq \int_{N_1(x)}^{\infty} \frac{(1+\epsilon)c_1}{u^\alpha}\, du + \cdots + \int_{N_m(x)}^{\infty} \frac{(1+\epsilon)c_m}{u^\alpha}\, du + \cdots + \int_{N_{n_0}(x)}^{\infty} \frac{\epsilon}{u^\alpha}\, du \\
&= \frac{1+\epsilon}{\alpha-1} \sum_{k=1}^{m} \frac{c_k}{(N_k(x))^{\alpha-1}} + \frac{\epsilon}{\alpha-1} \sum_{k=m+1}^{n_0} \frac{c_k}{(N_k(x))^{\alpha-1}}.
\end{aligned} \tag{28}$$

Furthermore, the greedy rule implies, for every $1 \leq k, l \leq n_0$, $k \neq l$,

$$\frac{q_{N_k(x)}^{(k)}}{s_k} \geq \frac{q_{N_l(x)+1}^{(l)}}{s_l},$$

which, in conjunction with (26) and (27), yields

$$\left(\frac{(1-\epsilon)c_l}{(1+\epsilon)c_1 s_l}\right)^{1/\alpha} N_1(x) - 1 \leq N_l(x) \leq \left(\frac{(1+\epsilon)c_l}{(1-\epsilon)c_1 s_l}\right)^{1/\alpha} N_1(x) + 1 \qquad l = 1, \ldots, m, \tag{29}$$

$$\left(\frac{(1-\epsilon)\epsilon}{c_1 s_l}\right)^{1/\alpha} N_1(x) - 1 \leq N_l(x) \leq \left(\frac{(1+\epsilon)\epsilon}{c_1 s_l}\right)^{1/\alpha} N_1(x) + 1 \qquad l = m+1, \ldots, n_0. \tag{30}$$

Hence, the total size of all documents stored in the cache can be bounded as

$$\begin{aligned}
x - \max_k s_k &\leq N_1(x) + s_2 N_2(x) + \cdots + s_{n_0} N_{n_0}(x) + \cdots + s_M N_M(x) \\
&\leq N_1(x) + s_2 \mu_2 N_1(x) + \cdots + s_{n_0} \mu_{n_0} N_1(x) + H,
\end{aligned} \tag{31}$$

where H is a large enough constant,

$$\mu_j \triangleq \left(\frac{(1+\epsilon)c_j}{(1-\epsilon)c_1 s_j}\right)^{1/\alpha} \quad \text{for } \; j = 1, \ldots, m \text{ and } \quad \mu_j \triangleq \left(\frac{(1+\epsilon)\epsilon}{c_1 s_j}\right)^{1/\alpha} \quad \text{for } \; j = m+1, \ldots, n_0. \tag{32}$$

Thus, (31) and (32) imply

$$N_1(x) \geq \frac{x - H}{1 + s_2\mu_2 + \cdots + s_{n_0}\mu_{n_0}}, \tag{33}$$

which, in combination with the lower bounds (29) and (30), leads to

$$N_k(x) \geq \frac{\mu_k(x-H)}{1+s_2\mu_2+\cdots+s_{n_0}\mu_{n_0}} \quad \text{for} \quad k=2,\ldots,n_0. \tag{34}$$

Therefore, after upperbounding (28) using the inequalities (33) and (34), multiplying them with $x^{\alpha-1}$, letting first $x \to \infty$ and then $\epsilon \downarrow 0$, we obtain

$$\limsup_{x\to\infty} x^{\alpha-1} P_{os}(x) \leq \frac{(c_1^{1/\alpha}+\cdots+c_m^{1/\alpha}s_m^{1-1/\alpha})^\alpha}{\alpha-1},$$

which, in conjunction with Lemma 2, proves the result. $\diamond$

The following easy consequence of the preceding results shows that the LRU-S algorithm is within a constant factor from the optimal static one.

Corollary 1 *The asymptotic ratio between the fault probabilities of the LRU-S and optimal static algorithm satisfies*

$$\lim_{x\to\infty} \frac{P_s(x)}{P_o(x)} = K(\alpha).$$

Proof: Follows from Theorems 2, 3 and 4. $\diamond$

Remark: The constant $K(\alpha)$ can be understood as a relative penalty that the LRU algorithm pays in relation to the optimal static one for occasionally placing the infrequent items in the cache. Once in the cache, these items can stay there for a long time and, therefore, lower the LRU relative performance. $\diamond$

4.2 Optimizing the latency of document retrieval

In the Web environment, it may be desirable to minimize the expected latency of document retrieval, instead of the cache fault probability. In a first approximation, one can assume that the time to deliver a document from the cache is basically zero, while the transmission time of a document of size s_k from its original location depends on its size and is equal to ϕ_k. Then, using the same notation as in Section 2, it can be shown that the stationary limiting (as $\mathbf{N} \to \infty$) expected delay $\Phi(x)$ satisfies

$$\Phi(x) = \mathbb{E}\left(\phi_{I_0} 1[S_{R_0}^{(I_0)}(T)+s_{I_0} > x]\right),$$

since $\{S_{R_0}^{(I_0)}(T)+s_{I_0} > x\}$ represents the event that the currently requested document (R_0, I_0) is not in the cache. Note that in a different framework, ϕ_k may stand for some other cost of not finding document of type k in the cache.

Now, by mimicking the proof of Theorem 2, it is easy to show that under Assumption 1,

$$\lim_{x\to\infty} \Phi(x)x^{\alpha-1} = \frac{K(\alpha)}{(\alpha-1)}\left(c_1^{1/\alpha}p_1^{1/\alpha}s_1+\cdots+c_m^{1/\alpha}p_m^{1/\alpha}s_m\right)^{\alpha-1}\left(\frac{\phi_1 c_1^{1/\alpha}}{p_1^{1-1/\alpha}}+\cdots+\frac{\phi_m c_m^{1/\alpha}}{p_m^{1-1/\alpha}}\right).$$

Then, similarly as in Theorem 3, it can be demonstrated that the performance of the randomized LRU scheme is optimized if the probabilities p_k are selected according to

$$p_k = d\frac{\phi_k}{s_k},$$

where $d = \min_j s_j/\phi_j$. In addition, this optimized algorithm is asymptotically within a constant $K(\alpha)$ from the corresponding optimal static arrangement. Since the derivation of these results involves a straightforward repetition of the preceding analysis, we omit the details. This algorithm that selects the randomization probabilities to be proportional to ϕ_k/s_k was also considered in [11]. Interestingly, when the latency ϕ_k for fetching a document from its original location is linearly proportional to the document size, it appears that the best policy is the traditional (non-randomized) LRU scheme.

5 Concluding remarks

In this paper we investigated a class of randomized LRU caching algorithms under the independent reference model with Zipf's law request probabilities that were recently empirically observed in the Web access patterns. These low complexity algorithms were recently proposed in [11] to mitigate the effect of variable document sizes on Web caching. For this class of algorithms, we provide an explicit asymptotic characterization of the cache fault probability as the cache size grows. Then, using this asymptotic result, we show that the randomized LRU algorithms achieve the best performance when the randomization probabilities are selected inversely proportional to document sizes; this algorithm is termed LRU-S. The relative gain of LRU-S in comparison to the ordinary LRU algorithm can be arbitrarily large, as pointed out in the remark after Theorem 3. Thus, the minor increase in implementation complexity is compensated well with potentially high improvement in performance. In addition, we show that LRU-S is within a constant factor from the optimal static algorithm.

Finally, we would like to point out that the majority of the Web access traffic is highly correlated and, therefore, the use of the independent reference model may be inappropriate. In this regard, we recently studied in [8] the performance of the traditional LRU system (same document sizes) in the presence of statistical correlation in the request sequence. We model the correlation using semi-Markov modulated processes that provide the flexibility for modeling strong statistical correlation, including the widely reported long-range dependence in the Web page request patterns. When the marginal frequency of requesting documents follows the generalized Zipf's law distribution, our main result from [8] shows that the cache fault probability is asymptotically, for large cache sizes, the same as in the corresponding LRU system with i.i.d. requests. Therefore, since the analysis of the randomized LRU is very similar to the one of the traditional LRU, it is almost certain that Theorems 2 and 3 hold for the class of, possibly highly correlated, semi-Markov modulated request sequences.

Acknowledgments

The authors thank the anonymous reviewers for their helpful comments.

Appendix

In this section we state three lemmas that are used in proving Theorem 2. Lemmas 4 and 5 correspond to Lemmas 1 and 2 of [7], while Lemma 6 is Lemma 4 of [8].

Lemma 4 *Assume that $q_i \sim c/i^\alpha$ as $i \to \infty$, with $\alpha > 1$ and $c > 0$. Then, as $t \to \infty$*

$$\sum_{i=1}^{\infty} (q_i)^2 e^{-q_i t} \sim \frac{c^{\frac{1}{\alpha}}}{\alpha} \Gamma\left(2 - \frac{1}{\alpha}\right) t^{-2+\frac{1}{\alpha}},$$

where Γ is the Gamma function.

Lemma 5 *Let $q_i \sim c/i^\alpha$ as $i \to \infty$, with $\alpha > 1$ and $c > 0$. Then, as $t \to \infty$*

$$\sum_{i=1}^{\infty}(1 - e^{-q_i t}) \sim \Gamma\left(1 - \frac{1}{\alpha}\right) c^{\frac{1}{\alpha}} t^{\frac{1}{\alpha}},$$

where Γ is the Gamma function.

Lemma 6 *Let $\{B_i, i \geq 1\}$ be a sequence of independent Bernoulli random variables, $S = \sum_{i=1}^{\infty} B_i$ and $m = \mathbb{E}[S]$. Then for any $\epsilon > 0$, there exists $\theta_\epsilon > 0$, such that*

$$\mathbb{P}[|S - m| > m\epsilon] \leq 2e^{-\theta_\epsilon m}.$$

References

[1] V. Almeida, A. Bestavros, M. Crovella, and A. de Oliviera. Characterizing reference locality in the WWW. In *Proceedings of the Fourth International Conference on Parallel and Distributed Information Systems*, Miami Beach, Florida, December 1996.

[2] P. Cao and S. Irani. Cost-aware WWW proxy caching algorithms. In *Proceedings of the USENIX 1997 Annual Technical Conference*, Anaheim, California, January 1997.

[3] J. A. Fill. An exact formula for the move-to-front rule for self-organizing lists. *Journal of Theoretical Probability*, 9(1):113–159, 1996.

[4] J. A. Fill and L. Holst. On the distribution of search cost for the move-to-front rule. *Random Structures and Algorithms*, 8(3):179–186, 1996.

[5] P. Flajolet, D. Gardy, and L. Thimonier. Birthday paradox, coupon collector, caching algorithms and self-organizing search. *Discrete Applied Mathematics*, 39:207–229, 1992.

[6] S. Irani. Page replacement with multi-size pages and applications to Web caching. *Algorithmica*, 33(1):384–409, 2002.

[7] P. R. Jelenković. Asymptotic approximation of the move-to-front search cost distribution and least-recently-used caching fault probabilities. *Annals of Applied Probability*, 9(2):430–464, 1999.

[8] P. R. Jelenković and A. Radovanović. Least-Recently-Used Caching with Dependent Requests. *Technical Report* EE2002-12-201, Department of Electrical Engineering, Columbia University, New York, August 2002; *Theoretical Computer Science*, to appear.

[9] P. R. Jelenković and A. Radovanović. Asymptotic Insensitivity of Least-Recently-Used Caching to Statistical Dependency. In *Proceedings of INFOCOM 2003*, San Francisco, April 2003.

[10] B. Moret and H. Shapiro. *Algorithms from P to NP: Volume 1 Design and Efficiency*. The Benjamin/Cummings Publishing Company, Redwood City, CA, 1991.

[11] D. Starobinski and D. Tse. Probabilistic methods for Web caching. *Performance Evaluation*, 46(2-3):125–137, 2001.

[12] N. Young. On-line file caching. *Algorithmica*, 33(1):371–383, 2002.