A New Upper Bound on Cache Hit Probability for Non-anticipative Caching Policies*

Nitish K. Panigrahy†, Philippe Nain‡, Giovanni Neglia § and Don Towsley†
† University of Massachusetts Amherst, MA, USA. Email: {nitish, towsley}@cs.umass.edu
‡ Inria, France. Email: philippe.nain@inria.fr
§ Inria, Sophia Antipolis, France. Email: giovanni.neglia@inria.fr

ABSTRACT

Caching systems have long been crucial for improving the performance of a wide variety of network and web based online applications. In such systems, end-to-end application performance heavily depends on the fraction of objects transfered from the cache, also known as the *cache hit probability*. Many cache eviction policies have been proposed and implemented to improve the hit probability. In this work, we propose a new method to compute an upper bound on hit probability for all non-anticipative caching policies, i.e. for policies that have no knowledge of future requests. At each object request arrival, we use hazard rate (HR) function based ordering to classify the request as a hit or not. Under some statistical assumptions, we prove that our proposed HR based ordering model computes the maximum achievable hit probability and serves as an upper bound for all non-anticipative caching policies. We also provide simulation results to validate its correctness and to compare it to Belady's upper bound. We find it to almost always be tighter than Belady's bound.

1. INTRODUCTION

Caches are pervasive in computing systems, and their importance is reflected in many networks and distributed environments including *content delivery networks* (CDNs). In such networks, the end user quality of experience primarily depends on whether the requested object is cached near the user. Thus the cache *hit probability*, i.e., the percentage of requests satisfied by the cache, plays an important role in determining the end-to-end application performance. In general, the number of objects available in a system is quite large compared to the cache capacity. Hence, the design of caching algorithms typically focuses on maximizing the overall cache hit probability. Also, maximizing the cache hit probability corresponds to minimizing the expected retrieval time, the load on the server and on the network for homogeneous setting.

One possible way to improve cache hit probability is by increasing cache capacity. However, increasing cache capacity only logarithmically improves cache hit probability[7]. Thus improving caching policies seems to be more effective for maximizing the overall cache hit probability. In practice, most caches employ *least-recently used* (LRU) or its variants often coupled with call admission or prefetching. Apart from LRU, other well known eviction policies include LFU, FIFO, RANDOM. There has been plethora of work [13, 18, 2, 4] on improving cache hit probabilities in the literature. In order to gauge the potential effectiveness of these eviction policies, an upper bound on maximum achievable cache hit probability for a given cache capacity has been widely adopted [2].

1.1 Offline upper bound

For equal size object, Belady's algorithm [1] has been widely used as an upper bound for cache hit probability among all feasible on demand and online caching policies, togetherly known as *non-anticipative* policies. However, Belady's algorithm is an offline algorithm, i.e., it assumes exact knowledge of future requests. Offline upper bounds on object hit probability have been proposed for variable size object [5]. Often system designers do not have access to the exact request trace, but can estimate the statistical properties of the object request process such as the *inter-request time* (irt) distribution. Also, caching studies typically include model driven simulations. Thus the following natural question arises. *With limited knowledge of the object arrival process and no look ahead option, can we provide an upper bound on the cache hit probability for any feasible online caching policy?*

1.2 Our Approach: Hazard Rate based upper bound

When object requests follow the *Independent Reference Model* (IRM), i.e. when objects are referenced independently with fixed probabilities, ideal Least-Frequently Used (LFU) caching policy is asymptotically optimal in terms of object hit probability. However, general request processes are more complex and correlated.

In this work, we assume a larger class of statistical models for object reference streams, see Section 2.1 and Section 2.2 for more details. We also assume that the *hazard rate* (HR) function (or conditional intensity) associated with this point process is well defined and can be computed at all points of time t. Here, the HR function is the conditional density of the occurrence of a object request at time t, given the

*This research was sponsored by the U.S. Army Research Laboratory and the U.K. Defence Science and Technology Laboratory under Agreement Number W911NF-16-3-0001 and by the NSF under grant NSF CNS-1617437. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Defence Science and Technology Laboratory.

Copyright is held by author/owner(s).

realization of the request process over the interval $[0, t)$ [9].

We now propose the HR based upper bound as follows. For each request arrival for object i at time t, we sort the objects according to the ratio of their HR values at time t to their sizes in decreasing order. Note that, an ideal LFU policy keeps track of number of times an object is referenced and order them accordingly in the cache. Similarly, in our upper bound, we keep an ordered list but on the basis of ratio of HR values to object sizes. We then classify the request as a hit if object i is among the B objects with the largest ratios. Here, B denotes the cache capacity.

Our contributions are summarized below:

1. We present a new upper bound for cache hit probability among all non-anticipative caching policies:
   - When objects have equal sizes, a simple HR based ordering for the objects provides an upper bound on cache hit probability.
   - For variable size objects, we order the objects with respect to the ratio of their HR function values to their objects sizes and provide upper bounds on the expected number of byte and object hits.
2. We evaluate and compare the HR based upper bound with different cache replacement policies for both synthetic and real world traces.

The rest of this paper is organized as follows. In Section 2 we formally present the HR based upper bound for equal size objects. In Section 3 we develop HR based upper bound for variable size objects. We perform simulation experiments to compare HR based upper bound with other policies in Section 4. Finally, the conclusion of this work and potential future works are given in Section 5.

2. EQUAL SIZE OBJECTS

We consider a cache of capacity B serving n distinct equal size objects. Without loss of generality we assume that all objects have size one. Later in Section 3, we also consider objects with different sizes. Let $\mathcal{D} = \{1, \cdots, n\}$ be the set of objects.

2.1 Number of Hits for General object Arrival Processes

Let $\{0 < T_{i,1} < T_{i,2} < \dots\}$ be the successive time epochs when object i is requested. For $k \geq 1$, define $X_{i,k} = T_{i,k} - T_{i,k-1}$, the inter-request time between the $(k-1)$st and kth request to object i, with $T_{i,0} = 0$ by convention.

Define $\tau_i(t) = \text{argmax}\{k \geq 1 : T_{i,k-1} < t\}$ so that exactly $\tau_i(t) - 1$ requests for object i are made in $[0, t)$. We denote by $\mathcal{H}_{t,i}$ the history of the request stream for object i up to time t, namely,

$$\mathcal{H}_{t,i} = \{T_{i,k}, k = 1, \dots, \tau_i(t) - 1\}.$$

The hazard rate function of $\{T_{i,k}, k = 1, 2, \dots\}$ at time t, is given by [9, Chapter 7]

$$\lambda_i^*(t) = \frac{\frac{d}{dt} P(X_{i,\tau_i(t)} < t - T_{i,\tau_i(t)-1} \,|\mathcal{H}_{t,i})}{P(X_{i,\tau_i(t)} > t - T_{i,\tau_i(t)-1} \,|\, \mathcal{H}_{t,i})}, \qquad (1)$$

by assuming the existence of $\frac{d}{dt} P(X_{i,\tau_i(t)} < t - T_{i,\tau_i(t)-1} \,|\mathcal{H}_{t,i})$ for each i and k (the latter is true if the point process $\{T_{i,k}\}_k$ is a regular point process [9]).

Under the assumption that the hazard rate functions $\lambda_i^*(t)$, $i = 1, \dots, n$ exist, the process $\{T_k\}_k$ resulting from the superposition of the point processes $\{T_{1,k}\}_k, \dots, \{T_{n,k}\}_k$ is a simple point process, namely, $0 < T_1 < T_2 < \dots$, since the probability that at least two objects are requested at the same time is zero. We define by $\mathcal{H}_t = \cup_{i=1}^n \mathcal{H}_{t,i}$ the history of the process $\{T_k\}_k$ up to time t. Call $R_k \in \{1, \dots, n\}$ the object requested at time T_k.

A caching policy π determines at any time t which B objects among the n available objects are cached. Formally, π is a measurable deterministic mapping from $\mathbb{R} \times (\times_{i=1}^n \mathbb{R}^\infty) \to S_B$, where S_B is the set of subsets of $\{1, \dots, n\}$ which contain B elements. In this setting, $\pi(t, \mathcal{H}_{t,1}, \dots, \mathcal{H}_{t,n})$ gives the B objects that are cached at time t.

Let Π be the collection of all such policies. Note that policies in Π are non-anticipative policies, in the sense that they do not know when future requests will occur.

We will only consider deterministic policies although the setting can easily be extended to random policies (in this case $\pi : \mathbb{R} \times (\times_{i=1}^n \mathbb{R}^\infty) \to \mathcal{Q}(S_B)$, where $\mathcal{Q}(S_B)$ is the set of probability distributions on S_B).

At time T_k there is a hit (resp. miss) if object R_k is in the cache (resp. is not in the cache).

We introduce the *hazard rate* (HR) based policy for equal-size objects, abbreviated as HR-E. At any time t and given $\mathcal{H}_t$, HR-E (i) determines the hazard rate function of each object and (ii) places in the cache the B documents that have the largest hazard rate functions, i.e. if $\lambda_{i_1}^*(t) > \dots > \lambda_{i_n}^*(t)$ then objects $i_1, \dots, i_B$ are cached at time t.

For any $\pi \in \Pi$, let $H_k^\pi = 1$ if the k-th object request is a hit and $H_k^\pi = 0$ otherwise. Let $N_K^\pi = \sum_{k=1}^K H_k^\pi$ be the number of hits during the first K requests for a object.

Let $B_k^\pi \in S_B$ be the state of the cache just before time T_k under π. Note that $B_k^{HR-E} = \pi(T_k-, \mathcal{H}_{T_k-})$. The following lemma holds, regardless of the state of the cache at time $t = 0$.

LEMMA 1 (EXPECTED NUMBER OF HITS).
Assume that the request object processes $\{T_{i,k}, k = 1, 2, \dots\}$, $i = 1, \dots, n$, are mutually independent. Then, for $K \geq 1$,

$$\mathbb{E}\left[N_K^{HR-E}\right] \geq \mathbb{E}\left[N_K^\pi\right], \quad \forall \pi \in \Pi. \qquad (2)$$

PROOF. See Appendix 7.1. □

It is worth noting that Lemma 1 holds for any non-stationary request object processes.

2.2 Upper Bound on the Hit Probability for Stationary and Ergodic object Arrival Processes

Let $\{\dots < T_{i,-1} < T_{i,0} \leq 0 < T_{i,1} < \dots\}$ be the successive time epochs when object i is requested. Define $X_{i,k} = T_{i,k} - T_{i,k-1}$ and introduce the two-sided sequence $\{X_{i,k}, k \in \mathbb{Z}\}$ of inter-request times to object i. We assume that $\{X_{i,k}, k \in \mathbb{Z}\}$ is a stationary and ergodic sequence, and that $X_{i,k}$ has a finite mean given by $\mathbb{E}[X_{i,k}] = \frac{1}{\lambda_i}$. We further assume that the point processes $\{T_{i,k}, k \in \mathbb{Z}\}$, $i = 1, \dots, n$, are mutually independent.

Define the point process $\{\dots < T_{-1} < T_0 < 0 \leq T_1 < T_2 < \dots\}$ obtained as the superposition of the n point processes $\{T_{i,k}, k \in \mathbb{Z}\}$, $i = 1, \dots, n$, Define $X_k := T_k - T_{k-1}$, so that $\{X_k, k \in \mathbb{Z}\}$ is the sequence of inter-request times for the point process $\{T_k, k \in \mathbb{Z}\}$. Call $R_k \in \{1, \dots, n\}$ the object requested at time T_k (see below).

The stationarity, ergodicity, and independence assumptions placed on point processes $\{T_{i,k}, k \in \mathbb{Z}\}$, $i = 1, \ldots, n$, imply that the sequence $\{(X_k, R_k), k \in \mathbb{Z}\}$ is stationary and ergodic (see e.g. [3, pp. 33-34]).

The stationary hit probability of policy $\pi \in \Pi$ is defined as

$$h^\pi = \lim_{K\to\infty} \frac{1}{K} \sum_{k=1}^{K} \mathbb{1}(R_k \in B_k^\pi) \quad \text{a.s.}, \tag{3}$$

when this limit exists, where B_k^π is the object of the cache just before time T_k under policy π (i.e. $B_k^\pi = \pi(T_k-, \mathcal{H}(T_k-))$ – see the definition of a policy in Section 2.1).

Define $Y_k^\pi = \mathbb{1}(R_k \in B_k^\pi)$. For any $\pi \in \Pi$, there exists a measurable mapping $\varphi^\pi : (R \times R)^\infty \to [0, 1]$ such that $Y_k^\pi = \varphi^\pi((T_j, R_j), j \le k-1)$, which shows that the sequence $\{Y_k^\pi, k \in \mathbb{Z}\}$ is stationary and ergodic (e.g. see [15, Thm p. 62]). The ergodic theorem then yields the stationary hit probability (see e.g. [14, Thm 1])

$$h^\pi = \mathbb{P}(R_k \in B_k^\pi) \tag{4}$$

under π. We are now in position to state and prove the main result of the paper.

THEOREM 1 (STATIONARY HIT PROBABILITY).

$$h^{HR-E} \ge \max_{\pi\in\Pi} h^\pi.$$

PROOF. See Appendix 7.2. □

REMARK 1. *The computation of the HR-E based upper bound does not require the simulation of any caching policy. At each request for a object, one can evaluate the HR values for all objects. One can then treat the request R_k as a hit if the hazard rate of R_k is among the top B hazard rates at time T_k.*

3. EXTENSION TO VARIABLE SIZE OBJECTS

We now assume object i has size $s_i \in \mathbb{R}^+$ for all $i \in \mathcal{D}$. Below we compute an upper bound on expected number of hits for bytes and objects when objects have variable size.

3.1 Number of byte hits and and fractional knapsack problem

The setting and assumptions are that of Section 2.1. Note that, in this setting, fractional caching (FC) is allowed. Denote $\Lambda_i(t)$ as

$$\Lambda_i(t) := \lambda_i^*(t).$$

For any $\pi \in \Pi$, we modify the definition of H_k^π and N_K^π for computing upper bound on byte hit ratios. Let H_k^π the number of bytes of object in the cache that can be used to satisfy the k-th request. Let $x_{i,k}$ denote the fraction of object i in the cache at the time of the k-th request. Then $H_k = s_k x_{i,k}^\pi$ if the request is for object i. Let $N_K^\pi = \sum_{k=1}^K H_k^\pi$ be the number of bytes of object served from the cache during the first K requests for a object.

Given that a request for a object is made at time T_k with the history $\mathcal{H}_{T_k}$, $\mathbb{E}[H_k^\pi]$ can be expressed as

$$\mathbb{E}[H_k^\pi] = \frac{\sum_{j=1}^n x_{i,k}^\pi s_i \Lambda_i(T_k)}{\sum_{j=1}^n \Lambda_j(T_k)}, \tag{5}$$

Our goal is to maximize $\mathbb{E}[H_k^\pi]$ subject to capacity constraint which can be solved by the following optimization problem.

$$\textbf{FC}(\boldsymbol{\Lambda_k}, \boldsymbol{s}, B)\textbf{:} \quad \max \quad \sum_{i=1}^n s_i x_i \Lambda_i(T_k) \tag{6a}$$

$$\text{subject to} \quad \sum_{i=1}^n s_i x_i \le B \tag{6b}$$

$$0 \le x_i \le 1, \quad i = 1, \cdots, n. \tag{6c}$$

Note that, the optimization problem maps to a Fractional Knapsack Problem (FKP). The optimal solution to FC($\boldsymbol{\Lambda}, \boldsymbol{s}, B$) can be obtained in polynomial time as follows [11].

Denote $g_i(T_k) = s_i\Lambda_i(T_k)/s_i = \Lambda_i(T_k)$. We order the objects according to

$$g_{i_1}(T_k) \ge g_{i_2}(T_k) \ge \cdots \ge g_{i_n}(T_k), \tag{7}$$

We call object i_a[1] as the critical object with

$$i_a = \min\left\{i_j : s_{i_1} + s_{i_2} + \cdots + s_{i_j} > B\right\}.$$

Denote $f_{i_a} = \left(B - \sum_{j=1}^{a-1} s_{i_j}\right)/s_{i_a}$. Under our proposed HR based byte hit rate upper bound for variable size objects (HR-VB), the object request R_k is considered as a full hit (of size s_{R_k}) if $R_k \in \{i_1, i_2, \cdots, i_{a-1}\}$. If $R_k = i_a$ then it is a f_{i_a} fractional hit. All other cases are considered as full misses.

Thus by definition $\mathbb{E}[H_k^{HR-VB}] \ge \mathbb{E}[H_k^\pi]$, $\forall \pi \in \Pi$. Summing both sides for $k = 1, \ldots, K$ and from the definition of N_K^π we get

$$\mathbb{E}\left[N_K^{HR-VB}\right] \ge \mathbb{E}\left[N_K^\pi\right], \quad \forall \pi \in \Pi. \tag{8}$$

3.2 Number of object hits and 0-1 knapsack problem

Again, the setting and assumptions are that of Section 2.1. Note that, in this setting, objects are assumed to be indivisible (IC), i.e., every object hit counts the same (i.e., a hit for a large 1GB object and hit for a small 10B object both count as a "hit"). We assume the definitions of H_k^π and N_K^π in Section 2.1 still hold. Given that a request for a object is made at time T_k with history $\mathcal{H}_{T_k}$, $\mathbb{E}[H_k^\pi]$ can be expressed as

$$\mathbb{E}[H_k^\pi] = \frac{\sum_{i\in B_k^\pi} \Lambda_i(T_k)}{\sum_{j=1}^n \Lambda_j(T_k)}, \tag{9}$$

Since objects are indivisible, $\mathbb{E}[H_k^\pi]$ can be maximized by solving a 0-1 knapsack problem (KP) defined as follows.

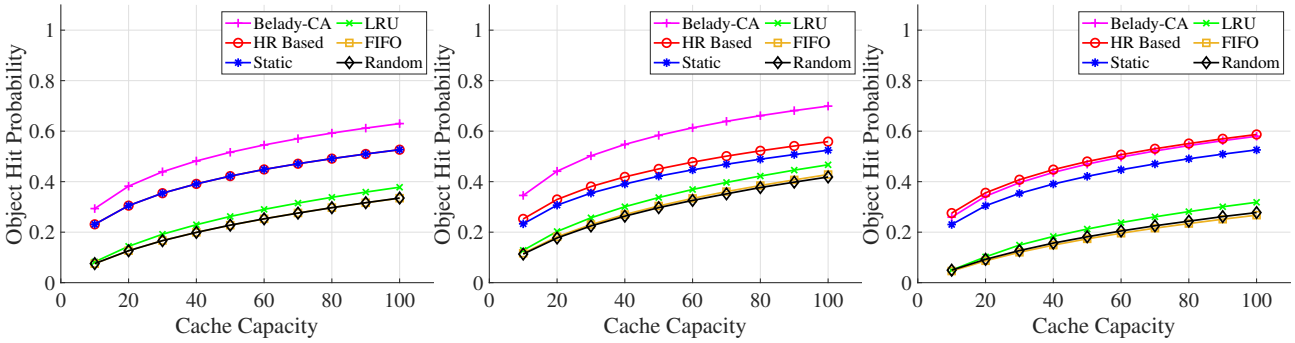
$$\textbf{IC}(\boldsymbol{\Lambda_k}, \boldsymbol{s}, B)\textbf{:} \quad \max \quad \sum_{i=1}^n x_i \Lambda_i(T_k) \tag{10a}$$

$$\text{subject to} \quad \sum_{i=1}^n s_i x_i \le B \tag{10b}$$

$$x_i \in \{0, 1\}, \quad i = 1, \cdots, n. \tag{10c}$$

Solving IC is NP-hard. However, the solution to the corresponding relaxed FKP serves as an upper bound to

[1] i_a is actually the critical object at time T_k. For brevity, we replace $i_a(T_k)$ with i_a.



(a) Exponential (b) Generalized-Pareto (c) Uniform

Figure 1: Simulation Results for HR based upper bound and various caching policies under different inter request arrival distributions.

Distribution	Hazard Rate	**Parameters**	$\mathbf{F_i(t)}$	Independent Parameters
Exponential	CHR	λ_i: rate	$1 - e^{-\lambda_i t}$	λ_i: Zipf(0.8)
Generalized Pareto	DHR	k_i : shape, σ_i: scale $\theta_i(=0)$: location	$1 - (1 + \frac{k_i t}{\sigma_i})^{-\frac{1}{k_i}}$	k_i : 0.48
Uniform	IHR	b_i : maximum value	t/b_i	$2/b_i$:Zipf(0.8)

Table 1: Properties of specific traffic distributions where IHR, DHR and CHR denote Increasing, Decreasing and Constant Hazard Rate respectively.

IP [11]. Thus we find an upper bound on $\mathbb{E}[H_k^\pi]$ by ordering the objects at time T_K according to (7), but with $g_i(T_k) = \Lambda_i(T_k)/s_i$. Define i_a and f_{i_a} as before. Following a procedure similar to that discussed in Section 3.1, we get the HR based upper bound on number of object hits for variable size objects (HR-VC). Thus by definition we have

$$\hat{H}_k^{HR-VC} = \frac{\sum_{j=1}^{a-1} \Lambda_{i_j}(T_k) + f_{i_a}\Lambda_{i_a}(T_k)}{\sum_{j=1}^{n} \Lambda_j(T_k)}$$
$$\geq \mathbb{E}[H_k^{HR-VC}] \geq \mathbb{E}[H_k^\pi], \ \forall \pi \in \Pi.$$

Summing the leftmost and rightmost expressions for $k = 1, \ldots, K$ and from the definition of N_K^π yields

$$\hat{N}_k^{HR-VC} \equiv \sum_{k=1}^{K} \hat{H}_k^{HR-VC} \geq \mathbb{E}\left[N_K^\pi\right], \quad \forall \pi \in \Pi.$$

4. EXPERIMENTS

We compare online policies for both equal and variable sized objects to that of our proposed upper bound as follows.

4.1 Equal Sizes

In our experiments, we consider a object catalog of size $n = 1000$. We vary cache size from $B = 10$ to $B = 100$. We assume that object popularities follow a Zipf distribution with parameter $\alpha = 0.8$. We consider various renewal point processes with different *inter-request time* (irt) distributions as shown in Table 1. The independent parameters for each irt distribution is shown in the last column of Table 1. The other parameters for each of the distribution are set such that the mean arrival rates (λ_i) across all the objects follow a Zipf distribution. We consider a half a million synthetic request trace and plot cache hit probability as a function of cache capacity. We compare the hit probability for different cache eviction policies to that of upper bounds as shown in Figure 1.

4.1.1 *Constant Hazard Rate*

When the request arrival process for each object has constant hazard rate (CHR), i.e the irts are described by an exponential distribution, the hit probability obtained for HR based upper bound equals that of STATIC policy, i.e. $h^{HR} = h^{STATIC}$ as shown in Figure 1(a). This is due to the fact that under CHR, $\Lambda_i(t) = \lambda_i \ \forall t \in [0, \infty)$ and $i \in \mathcal{D}$. Thus under HR, $B_k = \{\lambda_1, \lambda_2, \cdots, \lambda_B\}$ for all $T_k \in (0, \infty)$, which essentially is the STATIC policy.

4.1.2 *Decreasing Hazard Rate*

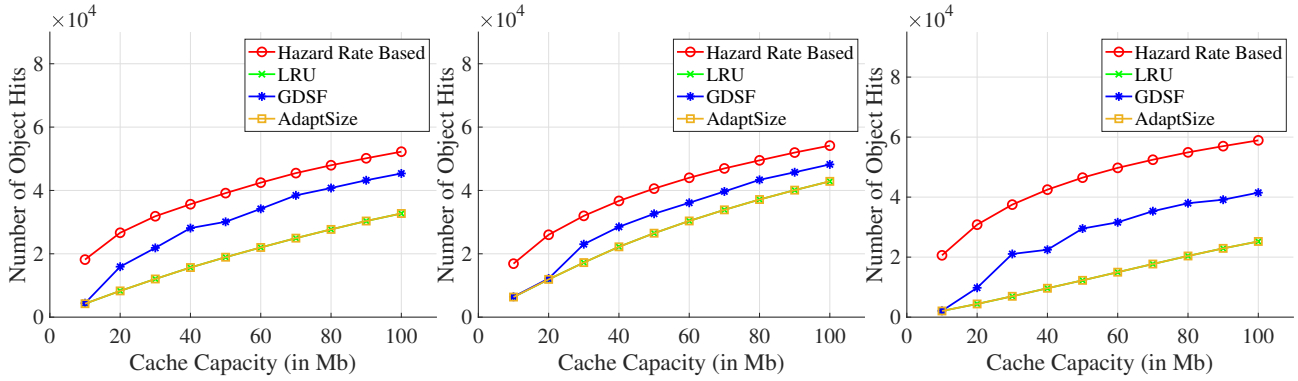
When the request arrival process for each object has a decreasing hazard rate (DHR), for ex: the irts are Pareto, HR based bound serves as an upper bound for different on-demand caching policies as shown in Figure 1 (b). Also, note that, BELADY-CA's policy produces a much larger hit probability compared to HR based bound, i.e. $h^{HR} << h^{BELADY-CA}$. Thus HR based bound is tighter and provides a performance benchmark compared to BELADY-CA under DHR. We get similar results for the case when the irts are Hyperexponential or Gamma distributed.

4.1.3 *Increasing Hazard Rate*

We present hit probability as a function of cache capacity for Uniform distributed irts in Figure 1 (c). Clearly, the HR based bound serves as an upper bound also for the case when irt distributions have increasing hazard rates (IHR). For both these distributions $h^{HR} \approx h^{BELADY}$. Note that, for uniform irts $h^{BELADY} < h^{HR}$. Note that, HR based upper bound can be realized as prefetching based upper bound. BELADY-CA only provides a provable upper bound for demand based caching policies. We get similar results for the case when the irts are Erlang distributed.

4.2 Variable Sizes

We consider the case when objects have variable sizes. For our simulation, we assume a object catalog of size $n = 100$.



(a) Exponential (b) Generalized-Pareto (c) Uniform

Figure 2: Simulation Results for HR based upper bound and various caching policies under different inter request arrival distributions for variable object sizes.

We vary cache size from $B = 10Mb$ to $100Mb$. We generate object sizes for each object independently according to a bounded Pareto distribution with Pareto shape parameter 1.8, minimum object size $5Mb$ and maximum object size $15Mb$.

We assume that object popularities follow a Zipf distribution with parameter $\alpha = 0.8$. We consider various renewal point processes such as Uniform, Pareto and Exponential distributions with parameters as shown in Table 1. We consider half a million synthetic request trace and plot cache hit probability as a function of cache capacity. We compare the HR based upper bound to that of different cache eviction/admission policies designed for variable object sizes. Examples include LRU, *Greedy Dual Size Frequency Caching Policy* (GDSF)[8] and AdaptSize [6].

We plot number of object hits as a function of cache capacity for various irt distributions as shown in Figure 2 (a)-(c). Note that our proposed HR based bound indeed serves as an upper bound for different online cache eviction policies. Also, note that, the number of object hits for conventional online caching policies (for example GDSF) is closer to our proposed upper bound under exponential and Pareto irt distributions (See Figures 2 (a) and (b)). However, the number of object hits for conventional caching policies are farther from the upper bound under uniform irt distribution. Thus our proposed upper bound is tighter when irt distributions have a non-increasing hazard rate. Another interpretation is that there may be room for improving caching policy performance in scenarios where irts have increasing hazard rates.

4.3 Real-world Data Trace and Hazard Rate Estimation

We now compare the hit probability for different cache eviction policies to HR-E under real data trace. We use requests from a web access trace collected from a gateway router at IBM research lab [20]. We filter the trace such that each object have been requested at least a hundred times. The filtered trace contains 3.5×10^6 requests with a object catalog of size $n = 5638$. We vary cache size from $B = 100$ to 600.

Various parametric and non-parametric estimators have been developed in the literature to estimate the hazard rate [19, 17]. Here, we adopt a parametric estimator model and assume that the inter-request times for each object are independent and identically distributed non-negative random variables. Note that the web and storage traffic inter-request times and access patterns are well modeled by the heavy-tailed distributions [10, 12]. Hence we fit the density of inter-request times of each object to a Generalized-Pareto distribution using the maximum likelihood estimation technique and estimate the hazard rate for each object accordingly.

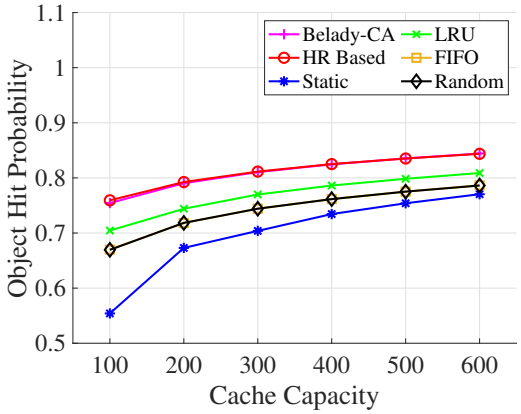


Figure 3: Performance comparison under real world data trace.

We compare HR-E to that of different cache eviction policies as shown in Figure 3. Note that, both BELADY-CA and HR-E provides an upper bound on cache hit probability for different online cache eviction policies. Also, note that, BELADY-CA's policy produces almost similar hit probabilities as that of HR-E.

5. CONCLUSION

In this paper, we developed an upper bound on the cache hit probability for non-anticipative caching policies with equal object sizes. We showed that hazard rate associated with the object arrival process can be used to provide this upper bound. Inspired by the results for equal size objects, we extended the HR based argument to obtain an upper bound on expected number of byte hits and object hits for variable size objects. We showed that HR based upper bound is tighter for a variety of object arrival processes than those analyzed in the literature. Future directions include to consider the

prefetching cost associated with any realizable hazard rate based caching policy.

6. REFERENCES

[1] A. V. Aho, P. J. Denning, and J. D. Ullman. Principles of Optimal Page Replacement. *J. ACM*, 18(1):80–93, 1971.

[2] M. Arlitt, L. Cherkasova, J. Dilley, R. Friedrich, and T. Jin. Evaluating content management techniques for web proxy caches. *Performance Evaluation Review*, 27(4):3–11, 2000.

[3] F. Baccelli and P. Brémaud. *Elements of Queueing Theory*. Springer, 2003.

[4] N. Beckmann, H. Chen, and A. Cidon. LHD : Improving Cache Hit Rate by Maximizing Hit Density Relative Size at. *NSDI*, 2018.

[5] D. S. Berger, N. Beckmann, and M. Harchol-Balter. Practical Bounds on Optimal Caching with Variable Object Sizes. *POMACS*, 2(2):1–32, 2018.

[6] D. S. Berger, R. K. Sitaraman, and M. Harchol-Balter. AdaptSize: Orchestrating the hot object memory cache in a content delivery network. *NSDI 2017*, pages 483–498, 2017.

[7] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker. Web caching and zipf-like distributions: Evidence and implications. *Proceedings - IEEE INFOCOM*, 1:126–134, 1999.

[8] L. Cherkasova. Improving WWW proxies performance with Greedy-Dual-Size-Frequency caching policy. *HP Laboratories Technical Report*, (98 -69), 1998.

[9] D. Daley and D. Vere-Jones. An Introduction to the Theory of Point Processes: Elementary Theory and Methods. *Springer*, 2003.

[10] A. B. Downey. Lognormal and pareto distributions in the internet. *Computer Communications*, 28(7):790 – 801, 2005.

[11] M. T. Goodrich and R. Tamassia. Chapter 5, The Fractional Knapsack Problem. *Algorithm Design: Foundations, Analysis, and Internet Examples*, pages 259–260, 2002.

[12] R. Gracia-Tinedo, Y. Tian, J. Sampé, H. Harkous, J. Lenton, P. García-López, M. Sánchez-Artigas, and M. Vukolic. Dissecting UbuntuOne: Autopsy of a global-scale personal cloud back-end. *IMC*, 2015-October(February):155–168, 2015.

[13] S. Jiang and X. Zhang. LIRS: An efficient low inter-reference recency set replacement policy to improve buffer cache performance. *Performance Evaluation Review*, 30(1):31–42, 2002.

[14] J. F. C. Kingman. The Ergodic Theory of Subadditive Stochastic Processes. *Journal of the Royal Statistical Society: Series B (Methodological)*, 30(3):499–510, 1968.

[15] P. Phillips. Lectures on stationary and nonstationary times series. 1992.

[16] S. Ross. *Stochastic Processes, 2nd Ed.* Wiley, 1996.

[17] N. D. Singpurwalla and M. Y. Wong. Kernel estimators of the failure-rate function and density estimation: An analogy. *Journal of the American Statistical Association*, 78(382):478–481, 1983.

[18] A. S. Tanenbaum. *Modern Operating Systems*. Prentice Hall Press, 2001.

[19] J.-L. Wang. Smoothing hazard rate. *Encyclopedia of Biostatistics (2nd ed.)*, 7:4986–4997, 2005.

[20] P. Zerfos, M. Srivatsa, H. Yu, D. Dennerline, H. Franke, and D. Agrawal. Platform and Applications for Massive-scale Streaming Network Analytics. *IBM Journal for Research and Development: Special Edition on Massive Scale Analytics*, 57(136):1–11, 2013.

7. APPENDIX

7.1 Proof of Lemma 1

PROOF. Given that a request for a content is made at time t and given that the history $\mathcal{H}_t$ is known, this request is for content i with the probability

$$p_i(t) = \frac{\lambda_i^*(t)}{\sum_{j=1}^n \lambda_j^*(t)}. \tag{11}$$

Proof of (11) is given as exercise 1.34 in [16]. This result relies on the mutual independence of the point processes $\{T_{i,k}, k = 1, 2, \ldots\}$, $i = 1, \ldots, n$. Observe that $p_i(t)$ does not depend on the caching policy in use.

By the definition of the HR-E policy

$$\sum_{i \in B_k^{HR-E}} \lambda_i^*(T_k-) \geq \sum_{i \in B_k^{\pi}} \lambda_i^*(T_k-), \tag{12}$$

for $k \geq 1$. Therefore for $k \geq 1$ (notice that under HR-E the content of the cache may change at time $t = 0+$)

$$\begin{aligned}
\mathbb{E}\left[H_k^{HR-E}\right] &= \frac{\sum_{i \in B_k^{HR-E}} \lambda_i^*(T_k-)}{\sum_{j=1}^n \lambda_j^*(T_k-)} \quad \text{from (11)}, \\
&\geq \frac{\sum_{i \in B_k^{\pi}} \lambda_i^*(T_k-)}{\sum_{j=1}^n \lambda_j^*(T_k-)} \quad \text{from (12)}, \\
&= \mathbb{E}\left[H_k^{\pi}\right], \qquad (13)
\end{aligned}$$

from (11) again. Summing both sides of (13) for $k = 1, \ldots, K$ gives (2) from the definition of N_K^{π}. □

7.2 Proof of Theorem 1

PROOF. Taking the expectation on both sides of (3), using the fact that h^{π} is a constant from (4), and then invoking the dominated convergence theorem, gives

$$h^{\pi} = \lim_{K \to \infty} \frac{1}{K} \sum_{k=0}^{K-1} \mathbb{E}[\mathbb{1}(R_k \in B_k^{\pi})] = \lim_{K \to \infty} \frac{\mathbb{E}[N_K^{\pi}]}{K}, \tag{14}$$

with $N_K^{\pi} = \sum_{k=0}^{K-1} \mathbb{1}(R_k \in B_k^{\pi})$ the number of hits in $[T_0, T_{K-1}]$ or, equivalently due to the stationary, the number of hits in K consecutive requests. Proof is concluded by using Lemma 1. □