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

Nitish K Panigrahy, Philippe Nain, Giovanni Neglia, Don Towsley

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

NITISH K. PANIGRAHY, University of Massachusetts Amherst
PHILIPPE NAIN, Centre Inria at Université Côte d'Azur
GIOVANNI NEGLIA, Centre Inria at Université Côte d'Azur
DON TOWSLEY, University of Massachusetts Amherst

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 transferred from the cache, also known as the *cache hit probability*. Many caching 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, for policies that have no knowledge of future requests. Our key insight is to order the objects according to the ratio of their *Hazard Rate* (HR) function values to their sizes and place in the cache the objects with the largest ratios till the cache capacity is exhausted. Under mild statistical assumptions, we prove that this cache allocation based on the HR-to-size ratio guarantees the maximum achievable hit probability for all non-anticipative caching policies. We derive closed form expressions for the upper bound under some specific object request arrival processes. We also provide simulation results to validate its correctness and to compare it to the state-of-the-art upper bounds, such as produced by Bélády's algorithm. We find it to be tighter than state-of-the-art upper bounds for a variety of object request arrival processes.

1 INTRODUCTION

Caches are pervasive in computing systems, and their importance is reflected in many networks and distributed environments including content delivery networks. 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 end-to-end application performance. Hence, the design of caching algorithms typically focuses on maximizing the overall cache hit probability. Also, when object sizes are equal, maximizing the cache hit probability corresponds to minimizing the expected retrieval time, the load on the server and on the network.

One possible way to improve cache hit probability is by increasing cache capacity. However, increasing cache capacity only logarithmically improves cache hit probability [2, 8]. 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 [6]. Apart from LRU, other well known eviction policies include *Least Frequently Used* (LFU), *First-In-First-Out* (FIFO), and RANDOM. There has been plethora of work [3, 5, 9, 16, 18, 22, 28] 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 [1].

1.1 Offline Upper Bound

For equal size objects, Bélády's algorithm or MIN [1] has been widely used as an upper bound for cache hit probability among all *non-anticipative* caching policies. However, Bélády'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 (different) size object [6]. 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 only statistical 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 non-anticipative 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, LFU 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 an object reference stream, 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 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 an object request at time t, given the realization of the request process over the interval $[0, t)$ [11].

We now propose the HR based upper bound. At any time t we determine the HR values of each object and place in the cache the objects that have the largest HR-to-size ratio till the cache capacity is exhausted. We emphasize that we do not provide new caching policies that outperform other policies but instead we provide upper bounds on cache hit probability.

Our contributions are summarized below:

(1) We present new upper bounds for cache hit probability among all non-anticipative caching policies, in particular,
   - an upper bound on the cache hit probability, for equal size objects;
   - upper bounds on the byte and object hit probabilities, for variable size objects.

(2) We derive closed form expressions for the upper bound under some specific object request arrival processes.

(3) 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 consider four specific object request arrival processes in Section 4. We perform simulation experiments to compare HR based upper bound with other policies in Section 5. Finally, we conclude the paper in Section 6.

2 EQUAL SIZE OBJECTS

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

2.1 Number of Hits for General object Arrival Processes

Let $\{0 < T_{i,1} < T_{i,2} < \cdots\}$ denote the successive time epochs when object i is requested. Assume $\{T_{i,k}\}_k$ is a regular point process, that is it possesses an intensity function [11, Definition 7.1.I., p.

213]. Define $X_{i,k} = T_{i,k} - T_{i,k-1}$ for $k \geq 2$ and $X_{i,1} = T_{i,1}$. For $t > 0$, define $\mathcal{H}_{i,t} = \{T_{i,k}, k \geq 1 : T_{i,k} < t\}$ to be the history of the point process $\{T_{i,k}\}_k$ in $[0, t)$.

Let $\{0 < T_1 < T_2 < \cdots\}$ be the point process resulting from the superposition of the point processes $\{T_{i,k}\}_k$, $i = 1, \ldots, n$. Call $R_k \in \{1, \ldots, n\}$ the object requested at time T_k. Define $\mathcal{H}_t = \{(T_k, R_k), k \geq 1 : T_k < t\}$ to be the history of point processes $\{T_{1,k}\}_k, \ldots, \{T_{n,k}\}_k$ in $[0, t)$. Notice that $\mathcal{H}_{i,t}$ is right-continuous for all i and so is $\mathcal{H}_t$. In particular, $T_k \notin \mathcal{H}_{T_k}$ for all k.

Define $k_i(t) = \max\{k \geq 1 : T_{i,k-1} < t\}$, so that exactly $k_i(t) - 1$ requests for object i have been made in $[0, t)$ (by convention $T_{i,0} = 0$).

Assume that the request object processes $\{T_{1,k}\}_k, \ldots, \{T_{n,k}\}_k$ are conditionally independent $\forall t > 0$, in the sense that

$$\mathbb{P}(T_{1,k_1(t)} \geq t_1, \cdots, T_{n,k_n(t)} \geq t_n \,|\, \mathcal{H}_t) = \prod_{i=1}^{n} \mathbb{P}(T_{i,k_i(t)} \geq t_i \,|\, \mathcal{H}_{i,t}), \tag{1}$$

for all $t_1 \geq t, \ldots, t_n \geq t$.

Given $T_{i,k} = t_{i,k}$ for $k \geq 1$, the hazard rate function of $\{T_{i,k}\}_k$ at time t is defined by the piecewise function [11, Definition 7.2.II, p. 231]

$$\lambda_i^*(t) = \begin{cases} \frac{\frac{d}{dt}P(X_{i,1} < t)}{P(X_{i,1} > t)}, & \text{for } 0 < t \leq t_{i,1}, \\ \frac{\frac{d}{dt}P(X_{i,k} < t - t_{i,k-1} \,|\, T_{i,j} = t_{i,j}, j \leq k-1)}{P(X_{i,k} > t - t_{i,k-1} \,|\, T_{i,j} = t_{i,j}, j \leq k-1)}, & \text{for } t_{i,k-1} < t \leq t_{i,k}, k \geq 2. \end{cases} \tag{2}$$

In (2) the existence of $\frac{d}{dt}P(X_{i,1} < t)$ and $\frac{d}{dt}P(X_{i,k} < t - t_{i,k-1} \,|\, T_{i,j} = t_{i,j}, j \leq k-1)$ for $k \geq 2$, follows from the assumption that $\{T_{i,k}\}_k$ is a regular point process [11, Definition 7.1.I., p. 213].

We assume that the cache is empty at time $t = 0$ to avoid unnecessary notational complexity but all results in the paper hold without this assumption as long as the probability distribution of the state of the cache is known at time $t = 0$. 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 $\mathbb{R} \times (\mathbb{R} \times \{1, \ldots, n\})^\infty \to S_B(n)$, where $S_B(n)$ is the set of subsets of $\{1, \ldots, n\}$ that contain B elements. In this setting, $\pi(t, \mathcal{H}_t)$ gives the B objects that are cached at time t based on the knowledge of the overall request process up to t. Let Π be the collection of all such policies. Note that policies in Π are *non-anticipative*, in 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 $\pi : \mathbb{R} \times (\mathbb{R} \times \{1, \ldots, n\})^\infty \to \mathcal{Q}(S_B(n))$, where $\mathcal{Q}(S_B(n))$ is the set of probability distributions on $S_B(n)$).

We introduce the hazard rate based *rule* for equal-size objects, abbreviated to 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 the B documents in the cache that have the largest hazard rates, i.e., if $\lambda_{i_1}^*(t) \geq \cdots \geq \lambda_{i_n}^*(t)$ then objects $i_1, \ldots, i_B$ are cached at time t (ties between equal rates are broken randomly). We call it a rule, not a policy and will use it as a way to upper-bound various performance metrics (see next)—which is the goal of this paper—regardless of whether it can be implemented.

Let $B_k^\pi \in S_B(n)$ be the state of the cache just before time T_k under π, and define

$$H_k^\pi = \mathbb{1}(R_k \in B_k^\pi), \tag{3}$$

i.e., $H_k^\pi = 1$ if the k-th requested object is in the cache under rule π, and $H_k^\pi = 0$ otherwise. Denote by

$$N_K^\pi = \sum_{k=1}^{K} H_k^\pi, \tag{4}$$

the number of hits during the first K requests for an object.

The following theorem holds,

Theorem 2.1 (Expected number of hits).

$$\mathbb{E}\left[N_K^{HR-E}\right] \geq \mathbb{E}\left[N_K^{\pi}\right], \quad \forall \pi \in \Pi, \ \forall K \geq 1. \tag{5}$$

Proof. Fix $\pi \in \Pi$. Given that a request for an object is made at time t and given that the history $\mathcal{H}_t$ is known, this request is for object i with probability

$$p_i(t) = \frac{\lambda_i^*(t)}{\sum_{j=1}^n \lambda_j^*(t)}. \tag{6}$$

Proof of (6) is given in Appendix 7.1. This result relies on the conditional independence of point processes $\{T_{1,k}\}_k, \ldots, \{T_{1,k}\}_k$, expressed in (1). By definition of HR-E,

$$\sum_{i \in B_k^{HR-E}} \lambda_i^*(T_k) \geq \sum_{i \in B_k^{\pi}} \lambda_i^*(T_k), \quad \forall k \geq 1. \tag{7}$$

Therefore, for $k \geq 1$,

$$\begin{aligned}
\mathbb{E}\left[H_k^{HR-E} \mid \mathcal{H}_{T_k}, T_k\right] &= \mathbb{P}\left(R_k \in B_k^{HR-E} \mid \mathcal{H}_{T_k}, T_k\right) = \sum_{i=1}^n \mathbb{P}(R_k \in B_k^{HR-E} \mid R_k = i) p_i(T_k) \\
&= \sum_{i=1}^n \mathbb{1}(i \in B_k^{HR-E}) \frac{\lambda_i^*(T_k)}{\sum_{j=1}^n \lambda_j^*(T_k)} \quad \text{from (6)}, \\
&= \sum_{i \in B_k^{HR-E}} \frac{\lambda_i^*(T_k)}{\sum_{j=1}^n \lambda_j^*(T_k)} \geq \sum_{i \in B_k^{\pi}} \frac{\lambda_i^*(T_k)}{\sum_{j=1}^n \lambda_j^*(T_k)} \quad \text{from (7)}, \\
&= \mathbb{E}\left[H_k^{\pi} \mid \mathcal{H}_{T_k}, T_k\right].
\end{aligned} \tag{8}$$

Taking expectation on both sides of (8) to remove the conditioning yields $\mathbb{E}\left[H_k^{HR-E}\right] \leq \mathbb{E}\left[H_k^{\pi}\right]$. Summing both sides of the last inequality for $k = 1, \ldots, K$ gives (5) from the definition of N_K^{π}. □

It is worth noting that Theorem 2.1 holds for any non-stationary request object process. We now study a more specific request arrival process and derive an upper bound on the object hit probability.

2.2 Upper Bound on Stationary Hit Probability

We still assume that $\{T_{1,k}\}_k, \ldots, \{T_{n,k}\}_k$ are regular processes and that the conditional independence assumption (1) holds. We define the stationary hit probability of any policy $\pi \in \Pi$ as

$$h^{\pi} = \lim_{K \to \infty} \frac{1}{K} \sum_{k=1}^{K} H_k^{\pi} \quad \text{a.s.}, \tag{9}$$

whenever this limit exists, where H_k^{π} is defined in (3).

We are now in position to state and prove the following result.

Theorem 2.2 (Stationary hit probability).

For any $\pi \in \Pi$, assume that the limit in (23) exists and that h^{π} is a constant. Then,

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

Proof. Let $\pi \in \Pi$. Taking the expectation on both sides of (23), using the fact that h^π is a constant and then invoking Lebesgue's dominated convergence theorem gives

$$h^\pi = \mathbb{E}[h^\pi] = \mathbb{E}\left[\lim_{K\to\infty} \frac{1}{K}\sum_{k=1}^{K} H_k^\pi\right] = \lim_{K\to\infty} \frac{1}{K}\sum_{k=1}^{K} \mathbb{E}[H_k^\pi] = \lim_{K\to\infty} \frac{\mathbb{E}[N_K^\pi]}{K}, \tag{10}$$

by using (4). The proof is concluded by using Theorem 2.1. □

Let us now discuss the existence of the limit in (23). When the inter-request time sequences $\{X_{1,k}\}_k, \ldots, \{X_{n,k}\}_k$ are stationary, ergodic and mutually independent, the sequence $\{(X_k, R_k)\}_k$ is stationary[1] and ergodic (see e.g., [4, pp. 33-34]). The latter result coupled with the fact that, for any $\pi \in \Pi$, there exists a measurable mapping $\varphi^\pi : (\mathbb{R} \times \{1, \ldots, n\})^\infty \to \{0, 1\}$ such that $H_k^\pi = \varphi^\pi((T_j, R_j), j \le k-1)$ (H_k^π is defined in (3)), shows that the sequence $\{H_k^\pi, k \in \mathbb{Z}\}$ is stationary and ergodic (e.g., see [26, Thm p. 62]). The ergodic theorem then ensures the existence of the limit in (23) and that (see e.g., [19, Thm 1])

$$h^\pi = \mathbb{P}(H_k^\pi), \quad \forall \pi \in \Pi. \tag{11}$$

Other instances when the limit in (23) exists and is a constant are discussed in Section 4.

3 VARIABLE SIZE OBJECTS

We now assume that object i has size $s_i, i \in \mathcal{D}$, expressed in bytes and that the capacity of the cache is B bytes.

3.1 Number of Byte Hits and Fractional Knapsack Problem

The setting and assumptions are those of Section 2.1 but we now allow *Fractional Caching* (FC)[2], i.e., we allow the storage of a fraction of an object in the cache instead of the entire object. We refer the interested reader to [21], [25] for literature on fractional caching. We denote by Π_{FC} the set of all non-anticipative FC policies.

For $\pi \in \Pi_{FC}$, let V_k^π denote the number of bytes served from the cache at the k-th request for an object and let $x_{i,k}^\pi$ denote the fraction of object i in the cache at the time of the k-th request. Then, $V_k^\pi = \sum_{i=1}^n s_i x_{i,k}^\pi \mathbb{1}(R_k = i)$. Let $W_K^\pi = \sum_{k=1}^K V_k^\pi$ denote the total number of bytes served from the cache during the first K requests for an object.

Let HR-VB denote the rule that at time t places *entirely* in the cache objects with the largest hazard rates until an object cannot fit; for the latter object only the number of bytes that needed to fill the cache are stored. All other objects are not cached.

The following theorem holds,

Theorem 3.1 (Expected number of byte hits).

$$\mathbb{E}[W_K^{HR-VB}] \ge \mathbb{E}[W_K^\pi], \quad \forall \pi \in \Pi_{FC},\ \forall K \ge 1.$$

Proof. Given a request for an object is made at time t and that the history $\mathcal{H}_t$ is known, we have already observed (see (6)) that this request is for object i with the probability $\lambda_i^*(t)/\sum_{j=1}^n \lambda_j^*(t)$.

[1] Let X_i be a random variable with the same distribution as that of the stationary inter-request time for object i. The stationary version of the inter-request times for object i is the sequence $\{X_{i,k}\}_k$ where $P(X_{1,k} < x) = 1/\mathbb{E}[X_i]\int_0^x \mathbb{P}(X_i > u)du$ and $X_{i,k} =_{\text{st}} X_i$ for $k \ge 2$.

[2] Fractional caching has its applications in large video object delivery systems where objects are composed of chunks stored independently.

Therefore,

$$\mathbb{E}[V_k^\pi \mid \mathcal{H}_{T_k}, T_k] = \sum_{i=1}^{n} \mathbb{E}[V_k^\pi \mid \mathcal{H}_{T_k}, T_k, R_k = i] \times \frac{\lambda_i^*(T_k)}{\sum_{j=1}^n \lambda_j^*(T_k)} = \frac{\sum_{i=1}^n s_i x_{i,k}^\pi \lambda_i^*(T_k)}{\sum_{j=1}^n \lambda_j^*(T_k)}. \tag{12}$$

Our goal is to find $\pi \in \Pi_{FC}$ that maximizes $\mathbb{E}[V_k^\pi \mid \mathcal{H}_{T_k}]$ subject to the capacity constraint on the size of the cache. This can be done by solving the optimization problem,

$$\max \quad \sum_{i=1}^{n} s_i x_i y_i \tag{13a}$$

$$\text{subject to} \quad \sum_{i=1}^{n} s_i x_i \leq B \tag{13b}$$

$$0 \leq x_i \leq 1, \quad i = 1, \cdots, n, \tag{13c}$$

with $y_i := \lambda_i^*(T_k)$ for $i = 1, \ldots, n$ and where T_k is fixed. This optimization problem is known as the *Fractional Knapsack Problem* (FKP) [14, Chapter 5.1]. Its solution depends on the respective values of the ratios $s_i y_i / s_i = y_i$ for $i = 1, \ldots, n$. Assume that $y_{i_1} \geq \cdots y_{i_n}$. Clearly, all elements of the permutation $i_1, \ldots, i_n$ of $1, \ldots, n$ depend on T_k. For this choice of $\{y_i\}_i$ or, equivalently, for this ordering of the hazard rates at time T_k, the solution to the problem (13) is given by $\hat{x}_{i_j,k} = 1$ for $1 \leq j \leq a := \max\left\{l : s_{i_1} + s_{i_2} + \cdots + s_{i_l} \leq B\right\}$, $\hat{x}_{i_{a+1},k} = (B - s_{i_1} - s_{i_2} - \cdots - s_{i_a})/s_{i_{a+1}}$ and $\hat{x}_{i_j,k} = 0$ for $j = a+2, \ldots, n$.

We have, by using the definition of $\{y_i\}$ and (12), that for any policy $\pi \in \Pi_{FC}$,

$$\mathbb{E}[V_k^{HR-VB} \mid \mathcal{H}_{T_k}, T_k] = \frac{\sum_{i=1}^n s_i \hat{x}_{i,k} \lambda_i^*(T_k)}{\sum_{j=1}^n \lambda_j^*(T_k)} \geq \frac{\sum_{i=1}^n s_i x_{i,k}^\pi \lambda_i^*(T_k)}{\sum_{j=1}^n \lambda_j^*(T_k)} = \mathbb{E}[V_k^\pi \mid \mathcal{H}_{T_k}, T_k].$$

Removing the conditioning on $\mathcal{H}_{T_k}$ and T_k yields

$$\mathbb{E}[V_k^{HR-VB}] \geq \mathbb{E}[V_k^\pi]. \tag{14}$$

Summing both sides of this inequality for $k = 1, \ldots, K$ gives

$$\mathbb{E}[W_K^{HR-VB}] \geq \mathbb{E}[W_K^\pi], \quad \forall K \geq 1. \tag{15}$$

□

Note that a similar reasoning can be found in [24], which studies caching policies to minimize the total miss cost, when a miss for object i incurs a generic cost c_i, under the simple IRM request process (for which hazard rates are constant over time). Our analysis focuses on $c_i = 1$ (hits) and $c_i = s_i$ (byte hits), but can be easily extended to such generic costs.

3.2 Upper Bound on the Byte Hit Probability for Stationary and Ergodic Object Arrival Processes

Throughout this section π is fixed in Π_{FC}, the set of fractional caching policies. We assume that object request arrival processes are stationary, ergodic and independent processes, so that the joint sequence of request arrival times and requested objects $\{(T_k, R_k)\}_k$ is stationary and ergodic (setting of Section 2.2). Recall that V_k^π is the number of bytes served by the cache at the kth request for an object. We denote by $\sigma_k = \sum_{i=1}^n s_i \mathbb{1}(R_k = i)$ the number of bytes requested from the cache at the kth request.

Define the stationary byte hit probability as

$$\widehat{h}^{\pi} = \lim_{K\to\infty} \frac{\sum_{k=1}^{K} V_k^{\pi}}{\sum_{k=1}^{K} \sigma_k}, \quad \text{a.s.,} \tag{16}$$

whenever this limit exists. The following theorem holds,

Theorem 3.2 (Stationary byte hit probability).

$$\widehat{h}^{HR-VB} \geq \widehat{h}^{\pi}, \quad \forall \pi \in \Pi_{FC}.$$

Proof. $\widehat{h}^{\pi}$ can be rewritten as

$$\widehat{h}^{\pi} = \lim_{K\to\infty} \frac{1}{K} \sum_{k=1}^{K} V_k^{\pi} \times \frac{1}{\lim_{K\to\infty} \frac{1}{K} \sum_{k=1}^{K} \sigma_k}, \quad \text{a.s.}$$

Since the sequence $\{R_k\}_k$ is stationary and ergodic the sequence $\{\sigma_k\}_k$ enjoys the same properties, and $\lim_{K\to\infty} \frac{1}{K} \sum_{k=1}^{K} \sigma_k = \mathbb{E}[\sigma]$ a.s., where σ is a rv with the same distribution as σ_k. On the other hand, there exists a mapping $\psi^{\pi} : (\mathbb{R} \times \{1, \ldots, n\})^{\infty} \to [0, \infty)$ such that $V_k^{\pi} = \psi^{\pi}((T_j, R_j), j \leq k)$, which shows that the sequence $\{V_k^{\pi}\}_k$ is stationary and ergodic [26, Thm p. 62]. Hence, $\lim_{K\to\infty} \frac{1}{K} \sum_{k=1}^{K} V_k^{\pi} = \mathbb{E}[V^{\pi}]$ a.s., with V^{π} a rv with the same distribution as V_k^{π}. Therefore,

$$\widehat{h}^{HR-VB} = \frac{\mathbb{E}[V^{HR-VB}]}{\mathbb{E}[\sigma]} \geq \frac{\mathbb{E}[V^{\pi}]}{\mathbb{E}[\sigma]} = \widehat{h}^{\pi}, \tag{17}$$

where the inequality follows from (14) since under the assumptions in Section 2.2 inequality (14) becomes $E[V^{HR-VB}] \geq \mathbb{E}[V^{\pi}]$ since the sequence $\{V_k^{\pi}\}_k$ is stationary and ergodic for all $\pi \in \Pi_{FC}$. □

3.3 Number of Object Hits and 0-1 Knapsack Problem

The setting and assumptions are those of Section 2.1, but we now assume that objects are indivisible, hereafter referred to as *Indivisible Caching* (IC). Moreover, we consider that 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"). Denote by Π_{IC} the set of all IC policies. Recall the definition of H_k^{π} (1 if hit at T_k and 0 otherwise) and N_K^{π} (number of hits in the first K requests) under $\pi \in \Pi_{IC}$—see (3) and (4), respectively.

Let HR-VC denote the following rule: at any time t, HR-VC places in the cache the objects in decreasing order of the ratios $\{\lambda_i^*(t)/s_i\}_i$ until an object does not fit in the cache. Assume that $\lambda_{i_1}^*(t)/s_{i_1} \geq \cdots \geq \lambda_{i_n}^*(t)/s_{i_n}$. If at time t object i_{l+1} is the first object that does not fit in the cache, then with probability $(B - s_{i_1} - s_{i_2} - \cdots - s_{i_l})/s_{i_{l+1}}$ this object is cached and with the complementary probability it is not cached. All subsequent objects i_j for $j > l + 1$ are not cached. Note that HR-VC does not meet the cache size constraint as there is not enough room in the cache to entirely fit object i_{l+1}. However, as mentioned in Section 2, our goal is to upper bound $\mathbb{E}[H_k^{\pi}]$ and $\mathbb{E}[N_k^{\pi}]$. The following theorem holds,

Theorem 3.3 (Expected number of object hits).

$$\mathbb{E}[N_K^{HR-VC}] \geq \mathbb{E}[N_K^{\pi}], \quad \forall \pi \in \Pi_{IC}, \ \forall K \geq 1.$$

Proof. Fix $\pi \in \Pi_{IC}$. We have by using (6)

$$\mathbb{E}[H_k^{\pi} \mid \mathcal{H}_{T_k} T_k] = \sum_{i=1}^{n} \mathbb{E}[H_k^{\pi} \mid \mathcal{H}_{T_k}, T_k, R_k = i] \times \frac{\lambda_i^*(T_k)}{\sum_{j=1}^{n} \lambda_j^*(T_k)} = \frac{1}{\sum_{j=1}^{n} \lambda_j^*(T_k)} \sum_{i=1}^{n} \mathbb{1}(i \in B_k^{\pi}) \lambda_i^*(T_k), \tag{18}$$

where we recall that B_k^π is the set of objects in the cache just before T_k under π.

It follows from (18) that $\mathbb{E}[H_k^\pi \mid \mathcal{H}_{T_k}]$ can be maximized by solving the following 0-1 *Knapsack Problem* (KP),

$$\max \quad \sum_{i=1}^{n} x_i y_i \tag{19a}$$

$$\text{subject to} \quad \sum_{i=1}^{n} s_i x_i \le B \tag{19b}$$

$$x_i \in \{0,1\}, \quad i = 1, \cdots, n, \tag{19c}$$

with $y_i := \lambda_i^*(T_k)$ for $i = 1, \ldots, n$ and T_k is fixed. Solving KP is NP-hard. However, it is known that the solution to the corresponding relaxed problem where the constraints in (19c) are replaced by $x_i \in [0,1]$ for $i = 1, \ldots, n$, serves as an upper bound for $\sum_{i=1}^n x_i y_i$ [14, Chapter 5.1]. Assume that the ordering $y_{j_1} \ge \cdots \ge y_{j_n}$ holds. With this ordering, the solution to the relaxed version of problem (19) is $\widetilde{x}^*_{j_l,k} = 1$ for $1 \le l \le b := \max\left\{m : s_{j_1} + s_{j_2} + \cdots + s_{j_m} \le B\right\}$, $\widetilde{x}^*_{j_{b+1},k} = (B - s_{j_1} - s_{j_2} - \cdots - s_{j_b})/s_{j_{b+1}}$ and $\widetilde{x}^*_{j_l,k} = 0$ for $l > b+1$ [14, Chapter 5.1]. Again, observe that all elements $j_1, \ldots, j_n$ of the permutation $1, \ldots, n$ depend on T_k. Denote by $\widetilde{x}_{1,k}, \ldots, \widetilde{x}_{n,k}$ the solution of (19). We know that

$$\sum_{i=1}^{n} \widetilde{x}^*_{i,k} y_i \ge \sum_{i=1}^{n} \widetilde{x}_{i,k} y_i. \tag{20}$$

Define the probability $p = (B - s_{j_1} - s_{j_2} - \cdots - s_{j_b})/s_{i_{b+1}}$, where the permutation $j_1, \ldots, j_n$ of $1, \ldots, n$ and b are defined in the paragraph above (20). By using the definition of $\{y_i\}$, we have,

$$\begin{aligned}
\mathbb{E}[H_k^{HR-VC} \mid \mathcal{H}_{T_k}, T_k] &= \frac{1}{\sum_{j=1}^n \lambda_j^*(T_k)} \left[\sum_{l=1}^{b} \lambda^*_{j_l}(T_k) + p\lambda^*_{j_{b+1}}(T_k) \right] \\
&= \frac{1}{\sum_{j=1}^n \lambda_j^*(T_k)} \sum_{i=1}^{n} \widetilde{x}^*_{i,k} \lambda_i^*(T_k) \\
&\ge \frac{1}{\sum_{j=1}^n \lambda_j^*(T_k)} \sum_{i=1}^{n} \widetilde{x}_{i,k} \lambda_i^*(T_k) \quad \text{from (20)}, \\
&\ge \frac{1}{\sum_{j=1}^n \lambda_j^*(T_k)} \sum_{i=1}^{n} \mathbb{1}(i \in B_k^\pi) \lambda_i^*(T_k) = \mathbb{E}[H_k^\pi \mid \mathcal{H}_{T_k}, T_k].
\end{aligned} \tag{21}$$

Removing the conditioning on $\mathcal{H}_{T_k}$ and T_k yields $\mathbb{E}[H_k^{HR-VC}] \ge \mathbb{E}[H_k^\pi]$. Summing both sides of this inequality for $k = 1, \ldots, K$ gives

$$\mathbb{E}[N_K^{HR-VC}] \ge \mathbb{E}[N_K^\pi]. \tag{22}$$

□

3.4 Upper Bound on Object Hit Probability for Stationary and Ergodic Object Arrival Processes

The setting is that in Section 2.2. Recall that the stationary hit probability of any policy $\pi \in \Pi_{IC}$ as

$$h^\pi = \lim_{K \to \infty} \frac{1}{K} \sum_{k=1}^{K} H_k^\pi \quad \text{a.s.}, \tag{23}$$

whenever this limit exists. We now prove the following theorem.

Theorem 3.4 (Stationary object hit probability).

$$h^{HR-VC} \geq h^{\pi}, \quad \forall \pi \in \Pi_{IC}.$$

Proof. Mimicking the arguments used to prove Theorem 2.2 in Section 2.2, one can show by using (22) that

$$h^{\pi} = \mathbb{E}[h^{\pi}] = \mathbb{E}\left[\lim_{K\to\infty} \frac{1}{K}\sum_{k=1}^{K} H_k^{\pi}\right] = \lim_{K\to\infty} \frac{1}{K}\sum_{k=1}^{K} \mathbb{E}[H_k^{\pi}] = \lim_{K\to\infty} \frac{\mathbb{E}[N_K^{\pi}]}{K}$$
$$\leq \lim_{K\to\infty} \frac{\mathbb{E}[N_K^{HR-VC}]}{K} = h^{HR-VC}, \tag{24}$$

for any policy π in Π_{IC}. □

4 SPECIFIC REQUEST ARRIVAL PROCESSES

Below we consider four specific object request processes each with equal size object, some of which explicitly account for temporal locality in requests for objects. Sections 4.1, 4.2, 4.4 (Poisson, shot noise, on-off) consider processes where requests to different objects are independent. Section 4.3 focuses on a *Markov Modulated Poisson Process* (MMPP) where requests to different objects are correlated.

4.1 Poisson Process

We consider the case where successive requests $\{T_{i,k}\}_k$ for object i ($i = 1, \ldots, n$) occur according to a Poisson process with rate $\lambda_i > 0$ and these n Poisson processes are mutually independent. Under these assumptions Theorem 2.2 in Section 2.2 applies from the discussion following the proof of this result.

This is the standard Independence Reference Model (see Section 1.2) where references to all objects are independent *random variables* (rvs). Without loss of generality assume that $\lambda_1 \geq \cdots \geq \lambda_n$. Under HR-E (see Section 2.1) objects $1, \ldots, B$ are always in the cache. Therefore, the hit probability h_i^{HR-E} for object i is $h_i^{\text{HR-E}} = \mathbb{1}(i \leq B)$ and the hit rate $r_i^{\text{HR-E}}$ for object i is $r_i^{\text{HR-E}} = \lambda_i \mathbb{1}(i \leq B)$. The overall hit probability $h^{\text{HR-E}}$ and hit rate r^{HR-E} are given by

$$h^{\text{HR-E}} = \frac{1}{\sum_{i=1}^{n} \lambda_i} \sum_{i=1}^{B} \lambda_i, \quad r^{\text{HR-E}} = \sum_{i=1}^{B} \lambda_i.$$

4.2 Shot Noise Model

The *Shot Noise Model* (SNM) [30] has been proposed to capture temporal locality observed in real traffic in caching systems e.g., in video on demand systems. The primary idea of the SNM is to represent the overall request process as the superposition of many independent time inhomogeneous Poisson processes or shots, each referring to an individual object. Each object i is characterized by three parameters τ_i, V_i and $\lambda_i(t)$, where τ_i is the time at which object i becomes available, V_i is the number of requests for object i, and $\lambda_i(t)$ is the popularity profile of object i. The mapping $\lambda_i(t)$ satisfies the following constraints: $\lambda_i(t) \geq 0$ for all $t \in \mathbb{R}$, $\lambda_i(t) = 0$ for $t < 0$ and $\int_0^{\infty} \lambda_i(t)dt = 1$. Conditioned on τ_i and V_i, the request process for object i is described by an inhomogeneous Poisson process with instantaneous rate at time t given by

$$\lambda_i^{\text{inst}}(t) = V_i \lambda_i(t - \tau_i). \tag{25}$$

The third constraint implies that the expected number of requests for object i, given $\int_0^{\infty} \lambda_i^{\text{inst}}(t)dt$, is equal to V_i. As in [30] we will assume in the experimental Section 5.1 that the time instants $\{\tau_i\}_i$

when objects become available form a Poisson process with rate γ and that V_i is drawn from a Poisson distribution.

The instantaneous hazard rate associated with object i is given by

$$\lambda_i^*(t) = \lambda_i^{\text{inst}}(t). \tag{26}$$

Note that, in the shot-noise model, there can potentially be an infinite number of objects in the catalog. However, at any time t, there is only a finite number of objects that are actively requested. Indeed, the number of active objects at time t is upper bounded by the number of occurrences in $[0, t)$ of the Poisson process $\{\tau_i\}_i$, which is always finite quantity.

Since object request processes are independent (as $\{\tau_i\}_i$ is a Poisson process) and inhomogeneous Poisson (and therefore regular) processes, Theorem 2.1 applies to the SNM. We conjecture that the limit in (23) exists and is a constant.

4.3 Markov Modulated Poisson Process

Let $\mathbf{Z} = \{Z(t), t \geq 0\}$ be a stochastic process taking values in a denumerable set $\mathcal{E}$. Throughout we assume that, given $Z(t) = x$, the object request processes $\{T_{1,k}\}_k, \ldots, \{T_{n,k}\}_k$ behave as independent Poisson processes with rate $\lambda_1(x), \ldots, \lambda_n(x)$, respectively, until the next jump of the process $\mathbf{Z}$. In other words, for each i, $\{T_{i,k}\}_k$ is an MMPP. However, since the conditional independence assumption (1) does not apply to the request processes $\{T_{1,n}\}_k, \ldots, \{T_{n,k}\}_k$ due to the correlations introduced by $\mathbf{Z}$, a specific analysis is required. It is done below.

We consider caching policies that may know the state of the environment at any time. Under the HR-E rule, if $Z(t) = x$ then the B objects with the largest arrival rates $\{\lambda_i(x)\}_i$ are stored in the cache at time t. The following theorem is the equivalent of Theorem 2.1 in Section 2.1.

Theorem 4.1. *For any policy $\pi \in \Pi$,*

$$\mathbb{E}[N_K^{\text{HR-E}}] \geq \mathbb{E}[N_K^{\pi}], \quad \forall K \geq 1.$$

Proof. The proof mimicks that of Theorem 2.1. Fix $k \geq 1$ and denote by $Z(T_k-)$ the state of the process $\mathbf{Z}$ just before time T_k, namely, just before the kth request for an object is made. For all $\pi \in \Pi$ and $x \in \mathcal{E}$, we have

$$\begin{aligned}
\mathbb{E}\left[H_k^{HR-E} \mid Z(T_k-) = x\right] &= \mathbb{P}\left(R_k \in B_k^{HR-E} \mid Z(T_k-) = x\right) \\
&= \sum_{i=1}^{n} \mathbb{P}\left(R_k \in B_k^{HR-E} \mid R_k = i, Z(T_k-) = x\right) \mathbb{P}(R_k = i \mid Z(T_k-) = x) \\
&= \sum_{i \in B_k^{HR-E}} \mathbb{P}(R_k = i \mid Z(T_k-) = x) \\
&= \sum_{i \in B_k^{HR-E}} \frac{\lambda_i(x)}{\sum_{j=1}^n \lambda_j(x)} \geq \sum_{i \in B_k^{\pi}} \frac{\lambda_i(x)}{\sum_{j=1}^n \lambda_j(x)} = \mathbb{E}\left[H_k^{\pi} \mid Z(T_k-) = x\right].
\end{aligned}$$

Removing the conditioning gives $\mathbb{E}\left[H_k^{HR-E}\right] \geq \mathbb{E}\left[H_k^{\pi}\right]$. □

We further assume that $\mathbf{Z}$ is an ergodic Markov process, so that the request processes are correlated Poisson processes. Denote by $(\theta(x), x \in \mathcal{E})$ the stationary distribution of $\mathbf{Z}$.

From now on the rule HR-E will be denoted by HR-MMPP to reflect the nature of the request processes.

The next result is the equivalent of Theorem 2.2 in Section 2.2. It shows that the HR-MMPP rule gives an upper bound for the stationary hit probability under any policy in Π.

THEOREM 4.2. *For any policy* $\pi \in \Pi$,

$$h^{\text{HR–MMPP}} \geq h^{\pi}.$$

PROOF. Under the assumptions made on the environment process (Markov process $\mathbf{Z}$) and the request processes (conditionally independent Poisson processes, modulated by $\mathbf{Z}$) the sequence $\{H_k^{\pi}\}_k$ (see (3)) is an ergodic sequence for any policy $\pi \in \Pi$, yielding the finiteness of the limit in (23) and the fact that this limit is a constant. The proof is then analogous to that of Theorem 2.2 in Section 2.2 by using the result in Theorem 4.1. □

Let us now calculate $h^{HR-MMPP}$. To avoid unnecessary complications, we assume from now on that the set $\mathcal{E}$ is finite.

Define the set $I(x)$ by $I(x) = \{i_1, \ldots, i_B\}$ if $\lambda_{i_1}(x) \geq \ldots \geq \lambda_{i_n}(x)$ where $i_1, \ldots, i_n$ is a permutation of $1, \ldots, n$ (if two or more objects have the same rate ties are broken randomly).

Define $\gamma(x) = \lim_{k\to\infty} \mathbb{P}(Z(T_k) = x)$, the stationary probability that the Markov process $\mathbf{Z}$ is in state x when a request for an object is made; for the time being assume that this limit exists.

We have (cf. Section 2.1),

$$\begin{aligned}
h^{HR-MMPP} &= \lim_{k\to\infty} \mathbb{P}(R_k \in B_k^{HR-MMPP}) \\
&= \lim_{k\to\infty} \sum_{x\in\mathcal{E}} \sum_{i=1}^{n} \mathbb{P}(R_k \in B_k^{HR-MMPP} \mid R_k = i, Z(T_k) = x)\mathbb{P}(R_k = i \mid X(T_k) = x)\mathbb{P}(Z(T_k) = x) \\
&= \lim_{k\to\infty} \sum_{x\in\mathcal{E}} \mathbb{P}(Z(T_k) = x) \sum_{i=1}^{n} \frac{\lambda_i(x)}{\sum_{j=1}^{n}\lambda_j(x)} \mathbb{P}(R_k \in B_k^{HR-MMPP} \mid R_k = i, Z(T_k) = x) \\
&= \lim_{k\to\infty} \sum_{x\in\mathcal{E}} \mathbb{P}(Z(T_k) = x) \sum_{i\in I(x)} \frac{\lambda_i(x)}{\sum_{j=1}^{n}\lambda_j(x)} = \sum_{x\in\mathcal{E}} \lim_{k\to\infty} \mathbb{P}(Z(T_k) = x) \sum_{i\in I(x)} \frac{\lambda_i(x)}{\sum_{j=1}^{n}\lambda_j(x)} \qquad (27) \\
&= \sum_{x\in\mathcal{E}} \gamma(x) \sum_{i\in I(x)} \frac{\lambda_i(x)}{\sum_{j=1}^{n}\lambda_j(x)}, \qquad (28)
\end{aligned}$$

where the interchange of the limit and the summation in (27) is justified by the finiteness of set $\mathcal{E}$.

It remains to calculate $\gamma(x)$. To this end, we use a standard Poisson uniformization technique which takes advantage of the fact that requests arrive according to a Poisson process with rate $\sum_{j=1}^{n}\lambda_j(x)$ when the Markov process $\mathbf{Z}$ is in state x. More specifically, let us sample $\mathbf{Z}$ according to a Poisson process with constant rate $\mu := \max_{x\in\mathcal{E}} \sum_{j=1}^{n}\lambda_j(x)$. Whenever there is an occurrence of the Poisson process and $\mathbf{Z}$ is in state x, this occurrence is selected with probability $\sum_{j=1}^{n}\lambda_j(x)/\mu$ and is not selected with the complementary probability. Therefore,

$$\gamma(x) = \frac{\mu \times \sum_{i=1}^{n}\lambda_i(x)/\mu \times \theta(x)}{\sum_{y\in\mathcal{E}} \mu \times \sum_{j=1}^{n}\lambda_j(y)/\mu \times \theta(y)} = \frac{\theta(x)\sum_{j=1}^{n}\lambda_j(x)}{\sum_{y\in\mathcal{E}}\theta(y)\sum_{j=1}^{n}\lambda_j(y)},$$

obtained as the ratio of the rate at which an occurrence of the Poisson process is selected when $\mathbf{Z}$ is in state x to the rate at which an occurrence of the Poisson process is selected.

Therefore, by (28),

$$h^{HR-MMPP} = \sum_{x\in\mathcal{E}} \frac{\theta(x)\sum_{i\in I(x)}\lambda_i(x)}{\sum_{y\in\mathcal{E}}\theta(y)\sum_{j=1}^{n}\lambda_j(y)}, \quad \forall x \in \mathcal{E}. \qquad (29)$$

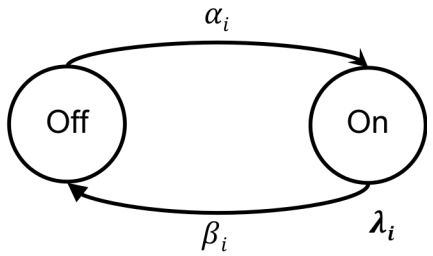


Fig. 1. On-Off Request Process

4.4 On-Off Request Process

Object popularity dynamics in caching systems can be effectively captured by using a stationary, on-off traffic model [13]. More specifically, we assume that successive requests to object i occur according to a Poisson process with rate $\lambda_i > 0$ when the underlying on-off process depicted in Figure 1 is in state 1 and no requests occur when this process is in state 0. The stationary distribution of this on-off process is given by $\boldsymbol{\pi}_i := [\pi_{i,0}, \pi_{i,1}] = [\beta_i/(\alpha_i+\beta_i), \alpha_i/(\alpha_i+\beta_i)]$. We assume that these n on-off processes are mutually independent. Under the above assumptions $\{T_{1,k}\}_k \ldots, \{T_{n,k}\}_k$ are independent and regular point processes, therefore fulfilling the assumptions in Section 2.1. In addition, the associated inter-request sequences $\{X_{1,k}\}_k, \ldots, \{X_{n,k}\}_k$ are independent, stationary and ergodic sequences, so that Theorem 2.2 in Section 2.2 applies thanks to the discussion following the proof of this result.

Without loss of generality, assume that $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$. Below, we derive expressions for the hit rate and hit probability under HR-E.

At any time, the HR-E rule stores the most popular objects in the cache. Without loss of generality, we assume that it only considers those whose associated on-off process are in state 1. Among them it places in the cache the objects with the largest request rate λ_i till the cache capacity is exhausted. Therefore, if $i > B$ the hit probability $h_i^{\text{HR-E}}$ for object i is given by

$$\begin{aligned} h_i^{\text{HR-E}} &= \mathbb{P}(\text{at most } B-1 \text{ on-off processes are in state 1 among on-off processes } 1,\ldots,i-1) \\ &= \sum_{k=0}^{B-1} \mathbb{P}(\text{exactly } k \text{ on-off processes are in state 1 among on-off processes } 1,\ldots,i-1) \\ &= \sum_{k=0}^{B-1} \sum_{\substack{i_1,\ldots,i_k \in \{1,2,\ldots,i-1\} \\ i_1<i_2<\cdots<i_k}} \prod_{l=1}^{k} \pi_{i_l,1} \prod_{m\in\{1,\ldots,i-1\}-(i_1,\ldots,i_k)} \pi_{i_m,0}, \end{aligned}$$

and $h_i^{\text{HR-E}} = 1$ if $i \leq B$. The hit rate $r_i^{\text{HR-E}}$ for object i is $r_i^{\text{HR-E}} = \lambda_i \pi_{i,1} h_i^{\text{HR-E}}$.

The overall hit probability $h^{\text{HR-E}}$ and the overall hit rate $r^{\text{HR-E}}$ are given by

$$h^{\text{HR-E}} = \sum_{i=1}^{n} \frac{\lambda_i \pi_{i,1}}{\sum_{j=1}^{n} \lambda_j \pi_{j,1}} h_i^{\text{HR-E}} \quad \text{and} \quad r^{\text{HR-E}} = h^{\text{HR-E}} \sum_{i=1}^{n} \lambda_i \pi_{i,1}. \tag{30}$$

Assume that $\pi_{i,0} = \rho$ for $i = 1, \cdots, n$. This occurs, for instance, if all n on-off processes have the same transition rates with $\alpha_i = \alpha$ and $\beta_i = \beta$ or if $\alpha_i = \alpha\theta_i$ and $\beta_i = \beta\theta_i$ for all i. Then, for $i > B$,

$$h_i^{\text{HR-E}} = \sum_{k=0}^{B-1} \sum_{\substack{i_1,\ldots,i_k \in \{1,2,\ldots,i-1\} \\ i_1<i_2<\cdots<i_k}} \rho^k (1-\rho)^{i-1-k} = (1-\rho)^{i-1} \sum_{k=0}^{B-1} \binom{i-1}{k} \left(\frac{\rho}{1-\rho}\right)^k,$$

so that

$$h^{\text{HR-E}} = \frac{\sum_{i=1}^{B}\lambda_i}{\sum_{i=1}^{n}\lambda_i} + \frac{1}{\sum_{i=1}^{n}\lambda_i}\sum_{i=B+1}^{n}\lambda_i(1-\rho)^{i-1}\sum_{k=0}^{B-1}\binom{i-1}{k}\left(\frac{\rho}{1-\rho}\right)^k,$$

and $r^{\text{HR-E}} = \rho\sum_{i=1}^{B}\lambda_i + \rho\sum_{i=B+1}^{n}\lambda_i(1-\rho)^{i-1}\sum_{k=0}^{B-1}\left(\frac{\rho}{1-\rho}\right)^k\binom{i-1}{k}$.

We now propose a recursive approach for computing the hit probability and the hit rate with a much lower computational complexity than the general formulas in (30).

The recursions are based on available objects in the catalog, starting from the situation where only object 1 is available, moving to the situation where objects 1 and 2 are available, etc. up to the final situation where all n objects are available. Let *occupancy* denote the number of objects in the cache and introduce the following variables,

$p_{l,k} = \mathbb{P}[\text{cache occupancy is } k \mid \text{catalog is composed of the } l \text{ most popular objects}]$,

$r_{l,k} =$ Hit rate when cache occupancy is k given catalog is composed of the l most popular objects .

When $l = 1$ then $p_{1,0} = \pi_{1,0}$, $p_{1,1} = \pi_{1,1}$, $r_{1,0} = 0$ and $r_{1,1} = \lambda_1$ from our convention that object i is not in the cache when on-off process i is in state 0. It is easy to verify that under HR-E the following recursions hold true for the occupancy probabilities,

$$\begin{aligned}
p_{l,0} &= p_{l-1,0}\pi_{l,0}, \quad l = 2,\cdots,n,\\
p_{k,k} &= p_{k-1,k-1}\pi_{k,1}, \quad k = 1,\ldots,B,\\
p_{l,k} &= p_{l-1,k-1}\pi_{l,1} + p_{l-1,k,}\pi_{l,0}, \quad 0 < k < \min(l,B), \quad l = 1,\cdots,n,\\
p_{l,B} &= p_{l-1,B-1}\pi_{l,1} + p_{l-1,B}, \quad l = B+1,\cdots,n.
\end{aligned} \tag{31}$$

Similarly, the following recursions hold true for the hit rates,

$$\begin{aligned}
r_{l,0} &= 0, \quad l = 2,\cdots,n,\\
r_{k,k} &= r_{k-1,k-1} + \lambda_k, \quad k = 2,\ldots B,\\
r_{l,k} &= \frac{p_{l-1,k-1}\pi_{l,1}(r_{l-1,k-1}+\lambda_l) + p_{l-1,k}\pi_{l,0}r_{l-1,k}}{p_{l-1,k-1}\pi_{l,1} + p_{l-1,k}\pi_{l,0}}, \quad 0 < k < \min(l,B), l = 1,\cdots,n,\\
r_{l,B} &= \frac{p_{l-1,B-1}\pi_{l,1}(r_{l-1,B-1}+\lambda_l) + p_{l-1,B}r_{l-1,B}}{p_{l-1,B-1}\pi_{l,1} + p_{l-1,B}}, \quad l = B+1,\cdots,n.
\end{aligned} \tag{32}$$

Once the above recursions have been solved, the overall hit rate $r^{\text{HR-E}}$ and hit probability $h^{\text{HR-E}}$ under HR-E are given by

$$r^{\text{HR-E}} = \sum_{k=1}^{B} p_{n,k}r_{n,k} \quad \text{and} \quad h^{\text{HR-E}} = \frac{r^{\text{HR-E}}}{\sum_{l=1}^{n}\lambda_l\pi_{l,1}}. \tag{33}$$

Remark 1. *The overall time complexity of implementing the recursive approach in* (31)-(32) *is* $O(nB)$.

5 NUMERICAL RESULTS

In this section we use simulations to compare the stationary object hit probabilities of various online policies (Section 5.1.1) to that of our proposed upper bound (*HR upper bound*), Bélády's upper bound (BELADY) and to a third bound (FOO, see Section 5.1.2).

The goals of this study are to *(i)* determine conditions under which HR provides tighter bound than other approaches, *(ii)* to determine how close different non-anticipative policies come to the HR bound, and *(iii)* the effect of the request process on these results. This study is done for a

number of requests processes (Section 5.1.3), for equal and different size objects (Section 5.1.4) and for several cache sizes. We first present the experimental setup and then discuss the results.

5.1 Experimental Setup

5.1.1 Investigated online policies. We consider several caching policies. The well-known LRU, FIFO and RANDOM cache replacement policies make space for a new object upon a miss by evicting the least recently used object, the least recently inserted one, and a random one, respectively. The STATIC policy keeps forever in the cache the B objects which have the largest average arrival rates. Notice that the HR based bound and the hit probability under STATIC are equal when successive requests for each object follow a Poisson process (Section 4.1). We also consider the *Greedy-Dual-Size-Frequency* (GDSF) policy [10] which combines recency with frequency and size to improve upon LRU. Last, the AdaptSize policy [7] uses an adaptive insertion probability preferring small sized objects (similarly to [17, 23, 24]).

5.1.2 Upper bounds on object hit probability. Aside our proposed HR based upper bound which applies to both equal and variable sized objects, two other upper bounds on the object hit probability proposed in literature are used, Bélády's offline upper bound (BELADY, Section 1.1) for equal sized objects and a *Flow Based Offline Optimal* (FOO) bound [6] for different sized objects. FOO upper bound is computed by representing caching as a min-cost flow problem. We also consider a practical flow-based offline optimal (PFOO) upper bound [6] for large synthetic traces.

5.1.3 Arrival process of object requests. In each plot in Figures 2–3, request processes for objects $i = 1, \ldots, n$ are independent renewal processes with Inter-Request Time distributions for object i (IRT_i) shown in Table 1. More specifically, in Figure 2(a) (resp. Figures 2(b)–2(f)) the request process for object $i = 1, \ldots, n$ has an exponential IRT (resp. Generalized Pareto, Uniform, Hyperexponential, Gamma, Erlang) with arrival rate λ_i drawn from a Zipf distribution with parameter 0.8 (see last column of Table 1); similarly, in Figure 3(a) (resp. Figures 3(b)–(c)) the IRT has an exponential (resp. Generalized Pareto, Uniform) distribution with arrival rate λ_i drawn from a Zipf distribution with parameter 0.8.

Inter-request time distribution (IRT)	Hazard Rate	$\mathbb{P}(IRT_i < t)$	Arr. rate λ_i (= $1/\mathbb{E}[IRT_i]$) drawn from Zipf(0.8)
Exponential	CHR	$1 - e^{-\lambda_i t}$	λ_i
Generalized Pareto	DHR	$1 - (1 + \frac{k_i t}{\sigma_i})^{-\frac{1}{k_i}}$, $k_i = 0.48$	$\frac{1-k_i}{\sigma_i}$
Hyperexponential*	DHR	$1 - \sum_{j=1}^{2} p_{ji} e^{-\theta_{j,i} t}$ $p_{1,i} + p_{2,i} = 1$ $p_{1,i}/\theta_{1,i} = p_{2,i}/\theta_{2,i} := \nu_i$ $SCV_i = \mathrm{var}(IRT_i)/\mathbb{E}[IRT_i]^2 = 2$	$\frac{1}{2\nu_i}$
Uniform	IHR	$\frac{t}{b_i}$	$\frac{2}{b_i}$
Gamma	DHR ($k_i < 1$)	$\frac{1}{\Gamma(k_i)}\gamma(k_i, \frac{t}{\theta_i})$, $k_i = 0.5$	$\frac{2}{\theta_i}$
Erlang	IHR	$\frac{\gamma(k_i, \mu_i t)}{(k_i-1)!}$, $k_i = 0.2$	$\frac{\mu_i}{2}$

Table 1. Inter-Request Time (IRT) distributions of the renewal request arrival processes in Figures 2–3 and their properties (CHR = Constant Hazard Rate, IHR = Increasing Hazard Rate, DHR = Decreasing Hazard Rate, SCV = Squared Coefficient of Variation). * $p_{1,i} = (1 - \sqrt{(SCV_i - 1)/(SCV_i + 1)})/2$ under the constraints in the 3rd column.

In Figures 4(a)–(b) the arrival request process for object i ($i = 1, \ldots, n$) is generated via an on-off process (see Section 4.4) and these n on-off processes are mutually independent. The transition rates for on-off process i are $\alpha_i = 1/T_{OFF}$ and $\beta_i = 1/T_{ON}$, with $T_{ON} = 7$ (days) and $T_{OFF} = 9T_{ON}$. The arrival rate λ_i in the on-state is given by $\lambda_i = V/T_{ON}$, where V is drawn from a Pareto distribution with pdf $f_V(v) = \beta V_{min}^{\beta}/v^{1+\beta}$, $\mathbb{E}[V] = 10$ and $\beta = 2$ [13].

In Figure 5 requests for objects are generated according to a two-state MMPP (see Section 4.3). Without loss of generality (w.l.o.g.), label these states 1 and 2. Let α and β be the state transition rate from state 1 to 2 and from state 2 to 1, respectively. The stationary state probabilities are $\gamma(1) = \beta/(\alpha+\beta)$ and $\gamma(2) = \alpha/(\alpha+\beta)$. In the simulations, we took $\alpha = 2\times10^{-3}$ and $\beta = 1.6\times10^{-3}$. In state j, successive requests for object i are generated according to a Poisson process with rate $\lambda_i(j)$ for $j = 1, 2$. In state 1, we assume that object arrival rates $\lambda_i(1), \ldots, \lambda_n(1)$ each follows a Zipf distribution with parameter 0.8; W.l.o.g assume that $\lambda_1(1) > \lambda_2(1) > \cdots > \lambda_n(1)$. In state 2, we assume that object arrival rates are given by $\lambda_i(2) = \lambda_{n+1-i}(1)$ for $i = 1, \cdots, n$.

In Figure 6 requests for objects are generated by the shot noise model described in Section 4.2. The parameters have been chosen to emulate the empirical trace 1 in [29, Table 2]. In particular, we consider requests for $c = 4$ classes. New objects are generated according to a homogeneous Poisson process with rate $\gamma = 4.110\times10^3$. The number of items generated for each class c (n_c) is set equal to the value observed in the empirical trace. Requests for the i-th object of class c start at time $\tau_i(c)$ and have intensity $\lambda_i^{\text{inst}}(t) = (V_i/\alpha_c)e^{-(t-\tau_i(c))/\alpha_c}$, with $\alpha_c = \overline{L}_c/\log_e 9$. $\overline{L}_c$ is the expected lifespan[3] of an object of class c and V_i is chosen according to a Poisson distribution with rate $\overline{V}_c$. Then, $\overline{V}_c$ is the expected number of requests of an object in class c. The values of $\overline{V}_c$ and $\overline{L}_c$ are set equal to the corresponding values in trace 1 as indicated in [29, Table 2] and are reported also in Table 2 for convenience.

Class id (c)	$\overline{L}_c$	$\overline{V}_c$	Catalog size (n_c)
Class 1	1.14	86.4	29481
Class 2	3.36	41.9	45570
Class 3	6.40	59.5	27435
Class 4	10.53	36.9	41385

Table 2. Parameters of the shot-noise models in Figure 6.

In Figure 7 we use requests from a Web access trace collected from a gateway router at IBM research lab [32]. We filter the trace such that each object has been requested at least a hundred times. The filtered trace contains 3.5×10^6 requests with an object catalog of size $n = 5638$. Various parametric and non-parametric estimators have been developed in the literature to estimate the hazard rate [27, 31]. 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 heavy-tailed distributions [12, 15]. 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.

[3]The expected lifespan of an object is defined as the time interval during which the total number of requests for an object changes from 10% to 90%. We refer interested readers to [29] for further details.

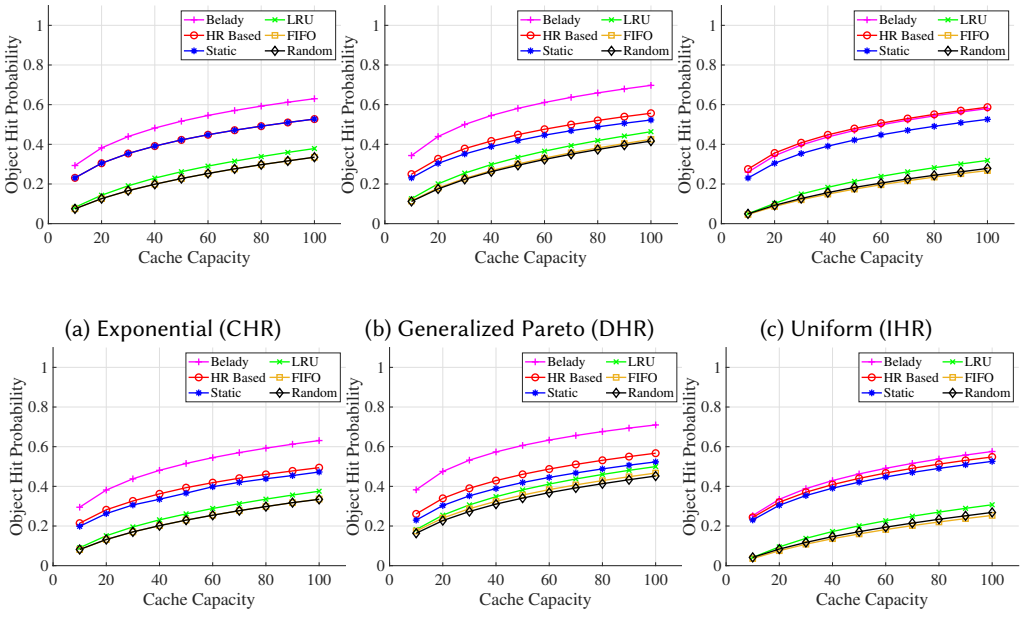


Fig. 2. Simulation results for HR based upper bound and various caching policies under different inter request arrival distributions ($n = 1000$, all objects of size 1).

5.1.4 Object sizes. Both objects of equal size and variable size are considered. In the former the size of each object is equal to one and in the latter the size of each object is drawn independently according to a bounded Pareto distribution with Pareto shape parameter 1.8, minimum object size of 5Mb and maximum object size of 15Mb. When all objects have the same size (resp. different size), the size of the cache is expressed in number of objects (resp. in Mb).

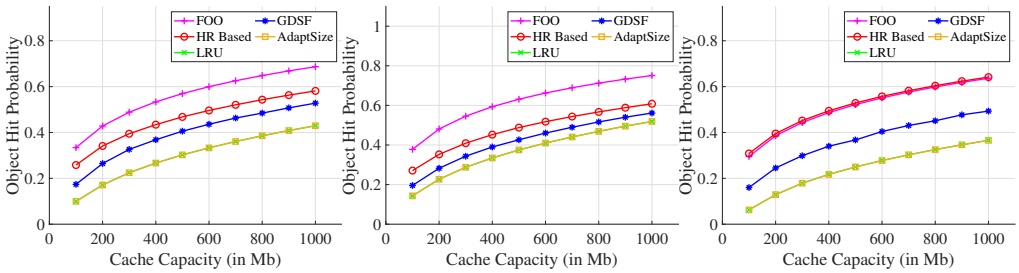
5.2 Discussion

As a general comment, Figures 2–7 confirm as expected that our HR model provides an upper bound on the hit probability among all online caching policies. Further comments are given below for each figure.

5.2.1 Renewal request processes and equal size objects. Request processes used to generate plots in Figure 2 are presented in Section 5.1.3. These plots are obtained for 1000 objects when all objects have size one. Notice (see discussion in Section 4.1) that results coincide in Figure 2(a) for the STATIC policy and the HR based upper bound when inter-request times (IRTs) are exponential distributed. We observe that when IRTs are either CHR or DHR, the HR based upper bound is much tighter than Bélády's upper bound and that both bounds are close when IRTs are IHR. STATIC consistently yields the highest hit probability and is always close to the HR upper bound. For exponential IRTs or, equivalently for independence reference model, STATIC is well known to be the optimal non-anticipative policy [20].

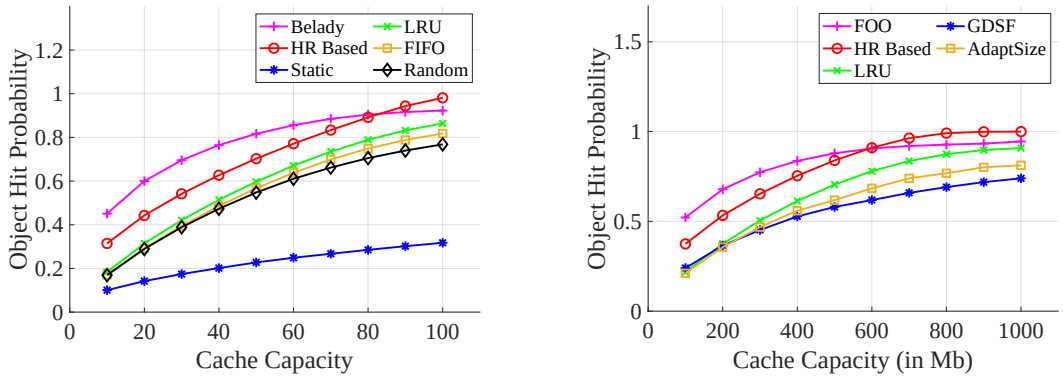
5.2.2 Renewal request processes and variable size objects. Request processes used to generate plots in Figure 3 are presented in Section 5.1.3. There are 1000 objects of variable size (see Section

5.1.4). We observe that when IRTs have exponential or Generalized Pareto distributions the HR based upper bound is much tighter than the FOO upper bound (Figure 3(a)–(b)) and that both bounds are close when IRTs are uniformly distributed rvs (Figure 3(c)). For exponential and Generalized Pareto IRT distributions the GDSF policy is close to HR and hence there is little room for improvement in caching policy performance. The gap between HR (resp. FOO) and GDSF in Figure 3(c) suggests that there is room for improvement in caching policy performance when IRTs are uniformly distributed rvs.



(a) Exponential (CHR) (b) Generalized Pareto (DHR) (c) Uniform (IHR)

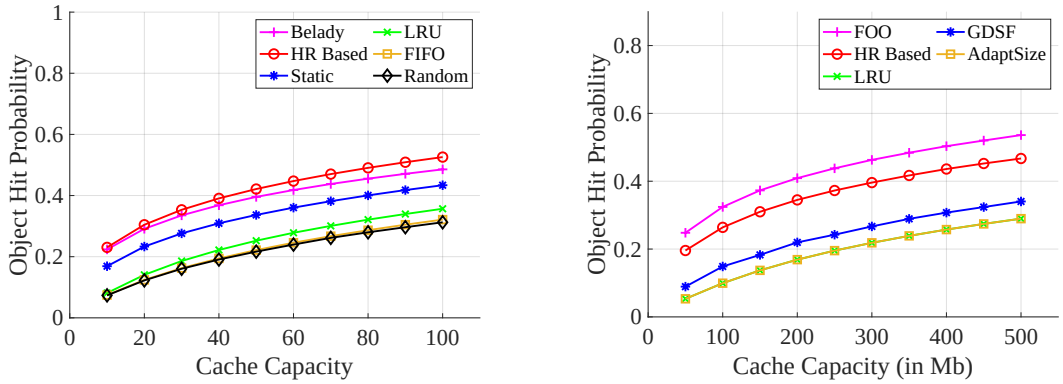
Fig. 3. Simulation Results for HR based upper bound and various caching policies under different inter request arrival distributions for variable object sizes ($n = 1000$).



(a) Equal Size ($n = 1000$) (b) Variable Size ($n = 1000$)

Fig. 4. Performance comparison under on-off request process

5.2.3 On-off request arrivals and equal/variable size objects. The parameters of the on-off process used to generate arrival times of requests of object i ($i = 1, \ldots, n$) are given in Section 5.1.3. There are 1000 objects in the catalog in both the equal sized and variable sized scenarios. The average arrival rate for object i is $\lambda_i \pi_{i,1}$ with $\pi_{i,1} = \alpha_i/(\alpha_i + \beta_i)$ (Section 4.4). STATIC permanently stores the B objects in decreasing order of $\{\lambda_i \pi_{i,1}\}_i$ in the cache.



(a) Equal Size ($n = 1000$)

(b) Variable Size ($n = 1000$)

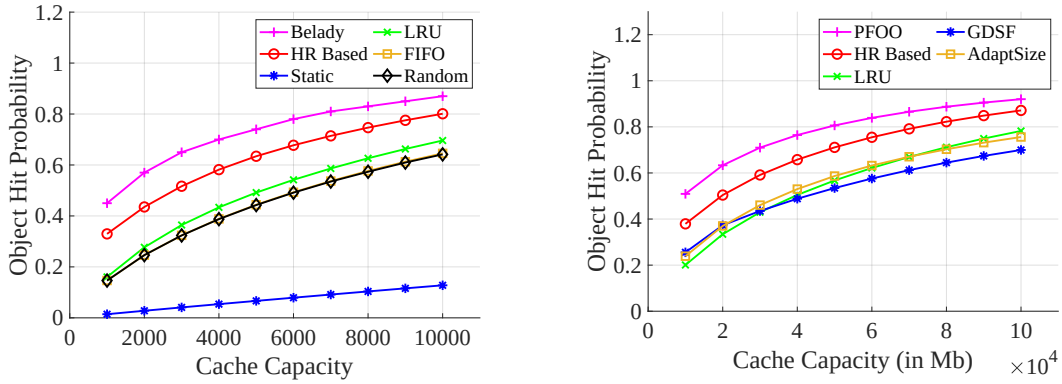
Fig. 5. Performance comparison under two-state MMPP request arrivals

For equal sized objects (resp. variable size objects) the HR bound is tighter than BELADY (resp. FOO) for small cache sizes whereas for larger cache sizes, BELADY (resp. FOO) becomes tighter. This may appear surprising at first, as BELADY has knowledge of the future request process, but our HR bound corresponds to the hit probability of an ideal policy that can prefetch objects in the cache as soon as they move to the on state, while BELADY bound corresponds to a reactive caching policy that performs optimal eviction decisions but cannot prefetch objects. We also observe that STATIC performs the worst for equal sized objects.

5.2.4 MMPP request arrivals and equal/variable size objects. The parameters of the two-state MMPP (states 1 and 2) used to generate arrival times of requests are given in Section 5.1.3. There are 1000 objects in the catalog in both the equal sized and variable sized scenarios. The average arrival rates for object i is $\lambda_i(1)\gamma(1) + \lambda_i(2)\gamma(2) = (\lambda_i(1)\beta + \lambda_i(2)\alpha)/(\alpha + \beta)$. STATIC permanently stores the B objects with the highest average arrival rates in the cache.

Unlike in Figures 2–4, BELADY is tighter than the HR based upper bound for equal sized objects (Figure 5(a)) but the latter upper bound is tighter than the FOO upper bound for variable sized objects (Figure 5(b)). STATIC is the closest to BELADY. Note that, in our simulations, $\gamma(1) = \beta/(\alpha + \beta)$ and $\gamma(2) = \alpha/(\alpha + \beta)$ are comparable. We observe that the performance of STATIC further improves when $\gamma(1) \gg \gamma(2)$ or $\gamma(1) \ll \gamma(2)$. For example, when $\gamma(1) \gg \gamma(2)$, $\lambda_i^{STATIC} \sim \lambda_i(1)\gamma(1)$; in this case the STATIC policy will permanently store the popular objects in state 1, thus always getting a hit when the MMPP is in state 1.

5.2.5 Shot noise request arrivals and equal/variable size objects. The parameters of the SNM used to generate Figure 6 are given in Section 5.1.3. For equal sized objects (Figure 6(a)) our proposed HR bound not only upper bounds the hit probability for existing online caching policies but also provides a tighter bound than the state-of-the-art BELADY. We also observed that STATIC obtained very small hit probabilities for different cache sizes. For variable sized objects, since the object catalog size and synthetic trace length was large, we considered a more practical (approximate) and efficient implementation of FOO upper bound, also known as Practical FOO (PFOO). We found our proposed HR upper bound to be tighter than the PFOO upper bound. The difference in the object



(a) Equal Size

(b) Variable Size

Fig. 6. Performance comparison under shot noise model (n = 143871).

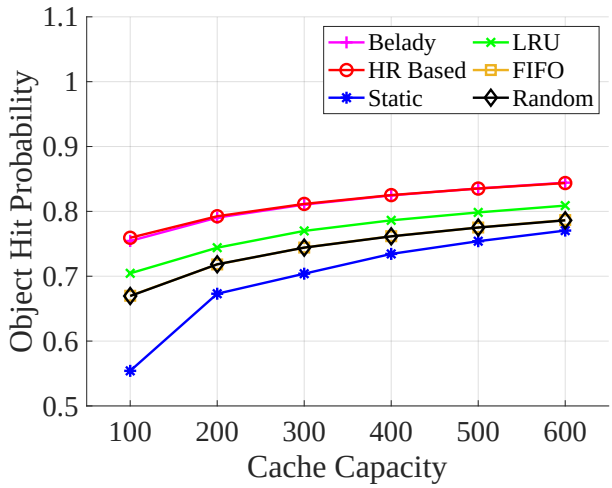


Fig. 7. Performance comparison under real world data trace (n = 5638).

hit probability between the HR upper bound and AdaptSize/LRU suggests that there is room for improvement in caching policy performance.

5.2.6 Real-world trace. Characteristics of the real-world trace and its application to the production of Figure 7 are discussed in Section 5.1.3. Upper bounds on the object hit probability obtained with HR and BELADY are almost identical. LRU performs the best and STATIC the worst.

6 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 the byte and object hit probability

for variable size objects solving a knapsack problem. We derived closed form expressions for the upper bound under some specific object request arrival processes. We showed that HR based upper bound is tighter than bounds proposed in the literature for a variety of object arrival processes.

REFERENCES

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

[2] Virgilio Almeida, Azer Bestavros, Mark Crovella, and Adriana de Oliveira. 1996. Characterizing Reference Locality in the WWW. In *4th IEEE Conf. on Parallel and Distributed Information Systems (PDSI'96)*. 92–103. DOI: https://doi.org/10.1109/pdis.1996.568672

[3] Martin Arlitt, Ludmila Cherkasova, John Dilley, Rich Friedrich, and Tai Jin. 2000. Evaluating content management techniques for web proxy caches, In ACM SIGMETRICS 2000. *Performance Evaluation Review* 27, 4, 3–11. DOI: https://doi.org/10.1145/346000.346003

[4] François Baccelli and Pierre Brémaud. 2003. *Elements of Queueing Theory*. Springer.

[5] Nathan Beckmann, Haoxian Chen, and Asaf Cidon. 2018. LHD : Improving Cache Hit Rate by Maximizing Hit Density Relative Size. In *NSDI'18*. 389–404.

[6] Daniel S. Berger, Nathan Beckmann, and Mor Harchol-Balter. 2018. Practical Bounds on Optimal Caching with Variable Object Sizes. *POMACS* 2, 2 (2018), 1–32.

[7] Daniel S. Berger, Ramesh K. Sitaraman, and Mor Harchol-Balter. 2017. AdaptSize: Orchestrating the Hot Object Memory Cache in a Content Delivery Network. In *NSDI'17*. 483–498.

[8] Lee Breslau, Pei Cao, Li Fan, Graham Phillips, and Scott Shenker. 1999. Web caching and zipf-like distributions: Evidence and implications. In *IEEE INFOCOM 1999*, Vol. 1. 126–134. DOI: https://doi.org/10.1109/INFCOM.1999.749260

[9] Pei Cao and Sandy Irani. 1997. Cost-aware WWW proxy caching algorithms. In *USENIX Symposium on Internet Technologies and Systems (USITS'97)*. 18. http://portal.acm.org/citation.cfm?id=1267297

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

[11] Daryl J. Daley and David Vere-Jones. 2003. An Introduction to the Theory of Point Processes: Elementary Theory and Methods. *Springer* (2003).

[12] Allen B. Downey. 2005. Lognormal and Pareto distributions in the Internet. *Computer Communications* 28, 7 (2005), 790 – 801. DOI: https://doi.org/10.1016/j.comcom.2004.11.001

[13] Michele Garetto, Emilio Leonardi, and Stefano Traverso. 2015. Efficient analysis of caching strategies under dynamic content popularity. In *IEEE INFOCOM 2015*. 2263–2271.

[14] Michael T. Goodrich and Roberto Tamassia. 2002. *Algorithm Design: Foundations, Analysis, and Internet Examples*. John Wiley & Sons.

[15] Raúl Gracia-Tinedo, Yongchao Tian, Josep Sampé, Hamza Harkous, John Lenton, Pedro García-López, Marc Sánchez-Artigas, and Marko Vukolic. 2015. Dissecting UbuntuOne: Autopsy of a global-scale personal cloud back-end. In *Internet Measurement Conference (IMC'15)*. 155–168. DOI: https://doi.org/10.1145/2815675.2815677

[16] Aamer Jaleel, Kevin B. Theobald, Simon C. Steely, and Jr Joel. 2010. High Performance Cache Replacement Using Re-Reference Interval Prediction (RRIP). In *37th Int. Symposium on Computer Architecture (ISCA'10)*. Saint Malo, France.

[17] Predrag R. Jelenković and Ana Radovanović. 2004. Optimizing LRU Caching for Variable Document Sizes. *Comb. Probab. Comput.* 13, 4fi?!5 (July 2004), 627fi?!643. DOI: https://doi.org/10.1017/S096354830400625X

[18] Song Jiang and Xiaodong Zhang. 2002. LIRS: An efficient low inter-reference recency set replacement policy to improve buffer cache performance, In ACM SIGMETRICS 2002. *Performance Evaluation Review* 30, 1, 31–42. DOI: https://doi.org/10.1145/511399.511340

[19] John F. C. Kingman. 1968. The Ergodic Theory of Subadditive Stochastic Processes. *Journal of the Royal Statistical Society: Series B (Methodological)* 30, 3 (1968), 499–510. DOI: https://doi.org/10.1111/j.2517-6161.1968.tb00749.x

[20] Zhen Liu, Philippe Nain, Nicolas Niclausse, and Don Towsley. 1998. Static Caching of Web Servers. In *Multimedia Computing And Networking (MCNC'98)*, SPIE Press (Ed.). San Jose, CA, USA.

[21] Lorenzo Maggi, Lazaros Gkatzikis, Georgios Paschos, and Jérémie Leguay. 2018. Adapting caching to audience retention rate. *Computer Communications* 116 (2018), 159–171.

[22] Nimrod Megiddo and Dharmendra S. Modha. 2003. ARC: A Self-Tuning, Low Overhead Replacement Cache. In *FAST'03: 2nd USENIX Conference on File and Storage Technologies*. 115–130.

[23] Giovanni Neglia, Damiano Carra, Ming Dong Feng, Vaishnav Janardhan, Pietro Michiardi, and Dimitra Tsigkari. 2016. Access-time aware cache algorithms. In *Proc. of ITC-28*.

[24] Giovanni Neglia, Damiano Carra, and Pietro Michiardi. 2018. Cache Policies for Linear Utility Maximization. *IEEE/ACM Transactions on Networking* 26, 1 (2018), 302–313. DOI: https://doi.org/10.1109/TNET.2017.2783623

[25] Georgios S. Paschos, Apostolos Destounis, Luigi Vigneri, and George Iosifidis. 2019. Learning to Cache with No Regrets. In *Proceedings - IEEE INFOCOM 2019*. 235–243.
[26] Peter C.B. Phillips. 1992. Lectures on Stationary and Nonstationary Times Series. (1992). http://korora.econ.yale.edu/phillips/teach/notes/1988-lectures.pdf.
[27] Nozer D. Singpurwalla and Man Yuen Wong. 1983. Kernel estimators of the failure-rate function and density estimation: An analogy. *J. Amer. Statist. Assoc.* 78, 382 (1983), 478–481. DOI: https://doi.org/10.1080/01621459.1983.10478000
[28] Andews S. Tanenbaum. 2001. *Modern Operating Systems.* Prentice Hall Press.
[29] Stefano Traverso, Mohamed Ahmed, Michele Garetto, Paolo Giaccone, Emilio Leonardi, and Saverio Niccolini. 2013. Temporal locality in today's content caching: Why it matters and how to model it. *Computer Communication Review* 43, 5 (2013), 5–12.
[30] Stefano Traverso, Mohamed Ahmed, Michele Garetto, Paolo Giaccone, Emilio Leonardi, and Saverio Niccolini. 2015. Unravelling the Impact of Temporal and Geographical Locality in Content Caching Systems. *IEEE Transactions on Multimedia* 17, 10 (2015), 1839–1854. DOI: https://doi.org/10.1109/TMM.2015.2458043
[31] Jane-Ling Wang. 2005. Smoothing Hazard Rate. *Encyclopedia of Biostatistics (2nd ed.)* 7 (2005), 4986–4997.
[32] Petros Zerfos, Mudhakar Srivatsa, Hao Yu, David Dennerline, Hubertus Franke, and Dakshi Agrawal. 2013. Platform and Applications for Massive-scale Streaming Network Analytics. *IBM Journal for Research and Development: Special Edition on Massive Scale Analytics* 57, 136 (2013), 1–11.

7 APPENDIX

7.1 Proof of Equation (6)

Recall that $k_i(t) = \max\{k \geq 1 : T_{i,k-1} < t\}$, so that exactly $k_i(t) - 1$ requests for object i have been made in $[0, t)$. We drop the argument t in $k_i(t)$ as no confusion may occur. We have

$$p_i(t) = \mathbb{P}\left(T_{i,k_i} < T_{j,k_j}, \forall j \neq i \,|\, \mathcal{H}_t, \min_{j=1,\ldots,n} T_{j,k_j} = t\right), \quad \forall i = 1, \ldots, n.$$

For $h > 0$,

$$\begin{aligned}
&\mathbb{P}\left(T_{i,k_i} \in (t, t+h), T_{j,k_j} > t+h, \forall j \neq i \,\middle|\, \mathcal{H}_t, \min_{j=1,\ldots,n} T_{j,k_j} \in (t, t+h)\right) \\
&= \frac{\mathbb{P}\left(T_{i,k_i} \in (t, t+h), T_{j,k_j} > t+h, \forall j \neq i \,|\, \mathcal{H}_t\right)}{\mathbb{P}\left(\min_{j=1,\ldots,n} T_{j,k_j} \in (t, t+h) \,|\, \mathcal{H}_t\right)} \\
&= \frac{\mathbb{P}(T_{i,k_i} \in (t, t+h) \,|\, \mathcal{H}_{i,t}) \prod_{\substack{1 \leq j \leq n \\ j \neq i}} \mathbb{P}(T_{j,k_j} > t+h \,|\, \mathcal{H}_{j,t})}{\mathbb{P}\left(\min_{j=1,\ldots,n} T_{j,k_j} \in (t, t+h) \,|\, \mathcal{H}_t\right)},
\end{aligned} \tag{34}$$

by using the conditional independence assumption in (1). Let us focus on the denominator in (34). It can be written as

$$\mathbb{P}\left(\min_{j=1,\ldots,n} T_{j,k_j} \in (t, t+h) \,|\, \mathcal{H}_t\right) = \sum_{j=1}^{n} \mathbb{P}(T_{j,k_j} \in (t, t+h) > t+h, T_{l,k_l} > t+h, \, \forall l \neq j \,|\, H_t) + f(h), \tag{35}$$

with $f(h) \to 0$ as $h \to 0$, since as $h \to 0$ there can be at least one rv located in $(t, t+h)$ among rvs $T_{1,k_1}, \ldots, T_{n,k_n}$ since these rvs are absolutely continuous (as they have a density under the

assumption that point processes $\{T_{1,k}\}_k, \ldots, \{T_{n,k}\}_k$ are regular). Therefore,

$$\begin{aligned}
&\mathbb{P}\left(T_{i,k_i} \in (t, t+h), T_{j,k_j} > t+h, \forall j \neq i \,\middle|\, \mathcal{H}_t, \min_{j=1,\ldots,n} T_{j,k_j} \in (t, t+h)\right) \\
&= \frac{\mathbb{P}(T_{i,k_i} \in (t, t+h) \,|\, \mathcal{H}_{i,t}) \prod_{\substack{1 \leq j \leq n \\ j \neq i}} \mathbb{P}(T_{j,k_j} > t+h \,|\, \mathcal{H}_{j,t})}{\sum_{j=1}^n \mathbb{P}(T_{j,k_j} \in (t, t+h), T_{l,k_l} > t+h, \forall l \neq j \,|\, H_t) + f(h)} \\
&= \frac{\mathbb{P}(T_{i,k_i} \in (t, t+h) \,|\, \mathcal{H}_{i,t}) \prod_{\substack{1 \leq j \leq n \\ j \neq i}} \mathbb{P}(T_{j,k_j} > t+h \,|\, \mathcal{H}_{j,t})}{\sum_{j=1}^n \mathbb{P}(T_{j,k_j} \in (t, t+h) \,|\, \mathcal{H}_{j,t}) \times \prod_{\substack{1 \leq l \leq n \\ l \neq j}} \mathbb{P}(T_{l,k_l} > t+h \,|\, \mathcal{H}_{l,t}) + f(h)} \qquad (36) \\
&= \frac{\frac{\mathbb{P}(T_{i,k_i} \in (t, t+h) \,|\, \mathcal{H}_{i,t})}{\mathbb{P}(T_{i,k_i} > t+h \,|\, \mathcal{H}_{i,t})}}{\sum_{j=1}^n \frac{\mathbb{P}(T_{j,k_j} \in (t, t+h) \,|\, \mathcal{H}_{j,t})}{\mathbb{P}(T_{j,k_j} > t+h \,|\, \mathcal{H}_{l,t})} + f(h)} = \frac{\lambda_i^*(t+h)}{\sum_{j=1}^n \lambda_j^*(t+h) + f(h)}, \qquad (37)
\end{aligned}$$

where (36) follows from (1) and (37) follows from (2). Letting $h \to 0$ in (37) gives (6).