Performance models for hierarchy of caches: Application to modern DNS caches

Sara Alouf, Nicaise Choungmo Fofack, Nedko Nedkov

Performance Models for Hierarchy of Caches: Application to Modern DNS Caches[☆]

Sara Alouf[a,*], Nicaise Choungmo Fofack[a,1], Nedko Nedkov[a,2]

[a]*Inria, 2004 Route des Lucioles, BP. 93 06902 Sophia Antipolis Cedex, France*

Abstract

This paper studies expiration-based caching systems in which caches assign a timer to each content they store and redraw the timer upon a cache miss. The modern Domain Name System (DNS) hierarchy is a valid application case and will be used throughout the paper. We introduce analytical models to study expiration-based caching systems based on renewal arguments. For polytree cache networks, we derive the cache performance metrics and characterize at each cache the aggregate request process, the thinning process and the miss process. A constant TTL policy is proved to maximize/minimize the hit probability if the requests' renewal function is concave/convex. We find that no distribution maximizes the hit probability anywhere in a network of caches. We validate our theoretical findings using real DNS traces (single cache and network cases) and via trace-driven simulations (network case).

Keywords: Analysis, expiration-based, cache networks, Domain Name System, Time-To-Live (TTL), real DNS trace validation, renewal theory

1. Introduction

In-network caching is a widely adopted technique to provide an efficient access to data or resources on a world-wide deployed system while ensuring scalability and availability. For instance, caches are integral components of the Domain Name System, the World Wide Web, Content Distribution Networks, the Information-Centric Network (ICN) architectures, or the recently proposed Dynamic Page Caching systems by Akamai Technologies. Many of these systems are hierarchical. The content being cached is managed through the use of expiration-based policies using a time-to-live (TTL) or replacement algorithms such the Least Recently Used (LRU), First-In First-Out (FIFO), Random, etc.

In this paper, we focus on hierarchical systems that rely on expiration-based policies to manage their caches. These policies have the advantage of being fully configurable and provide parameters (i.e. timers) to optimize/control the network of caches. Each cache in the system maintains for each item a timer that indicates its duration of validity. This timer can be initially set by an external actor or by the cache itself only when a request yields a cache miss.

The Domain Name System (DNS) is a valid application case. In short, the DNS maintains in a distributed database the mappings, called *resource records*, between names and addresses in the Internet. Servers in charge of managing a mapping are said to be *authoritative*. Caches—used to avoid generating traffic up in the DNS hierarchy—are found in both servers and clients (devices of end users). Caching is however limited in duration to avoid having stale records which may break the domains involved.

[☆]This manuscript is an extended version of [1].

[*]Corresponding author

Email addresses: sara.alouf@inria.fr (Sara Alouf), nchoungmofofack.ext@orange.com (Nicaise Choungmo Fofack), nedkostefanovnedkov@gmail.com (Nedko Nedkov)

[1]This author is currently working at INGIMA.

[2]This author's contribution was made during a 4-month internship at Inria in 2014.

DNS cache updates are strongly related with how the DNS hierarchy works. When a requested resource record R is not found at the client's cache, the client issues a request to a bottom-level DNS server (usually that of the Internet server provider). If R cannot be resolved locally and is not found in the cache, the latter server forwards the request to a server higher in the hierarchy. The process repeats itself until R is fetched at a cache or ultimately from the disk of an authoritative server. The server providing R is called the *answerer*. The record R is sent back to the client through the reverse path between the answerer and the client, and a copy of R is left at each cache on this path.

According to RFC 6195, all copies of R are marked by the answerer with a time-to-live (TTL) which indicates to caches the number of seconds that their copy of R may be cached. Consequently, all copies of a record along a path would be cached mainly for the same duration. This RFC specification is called the *TTL rule* in the literature. Caches compliant with it are referred to as *traditional DNS caches*. Those overriding the advocated TTL with a locally chosen value (see [2, 3]) are called *modern DNS caches* [4].

In a tree of traditional DNS caches a request occurring anywhere just after the content expired in the local cache yields cache misses at all caches along the path to an authoritative server. Such a *miss synchronization effect* [5] is avoided with modern caches. Other differences between traditional and modern DNS caches can be found in the companion technical report [6].

The objective of this paper is to assess the performance of polytree of caches. Our contributions are: (i) we are the first to provide analytic models to study both a single (modern) DNS cache and a polytree of caches with general caching durations; (ii) we characterize the distribution of the DNS traffic flowing upstream in the DNS hierarchy besides deriving the usual cache performance metrics; (iii) we identify conditions under which the deterministic caching duration maximizes/minimizes the performance metrics; (iv) for the case of a polytree of caches we recursively characterize the request and miss processes at each cache and find closed-form expressions when a specific distribution for requests and caching durations is used; (v) we check the robustness of our single cache model over DNS traces collected at Inria and (vi) the robustness of our network of caches model through event/trace-driven simulations.

This paper extends our previous work [1] as follows. The performance metrics (hit and occupancy probabilities) of a single cache are also derived when requests are described by stationary and ergodic processes (see Proposition 1). As a direct consequence, the exact analysis derived on linear cache networks is extended to a large class of hierarchical cache networks called *linear star* networks which include linear and two-level tree/star networks (see Section 5.1). Closed-form expressions for cache consistency measures are provided under the assumption that contents requests and updates occur according to two independent renewal processes (see Section 4.1.2). The conditions under which the deterministic policy optimizes the metrics are new. The network analysis on trees is extended to polytrees in Section 5.4. Section 5.5 has been revised to fully derive the closure properties of the class of distributions called *diagonal matrix-exponential* (or diag.ME for short). A major extension with respect to [1] appears in Section 6 where we evaluate our model. Additional results using traces of Inria's DNS server and our model are provided. Also, trace-driven simulations on a binary tree cache network complement the previous event-driven ones and show that our model is still robust under real traffic conditions.

The rest of the paper is organized as follows. Section 2 reviews the works most relevant to this paper. Section 3 presents the scenario considered and some introductory material. Our single cache model is analyzed in Section 4 and the case of a tree of caches in Section 5. We validate our models in Section 6 and show some numerical results. Section 7 summarizes our findings.

2. Related Works

Since the recent observation of the modern behavior of DNS caches [4, 2], only few results of the state of the art are applicable to modern DNS caches. Hou *et al.* consider in [7] a tree of traditional DNS caches fed by Poisson traffic. The performance metrics derived in [7] cannot characterize modern caches as these do not cause a miss synchronization effect—like traditional caches do—which is extensively used in their model.

Jung, Berger and Balakrishnan study in [8] a single traditional DNS cache fed by a renewal process. Their model assumes that each content is cached for a deterministic duration which would be either the

value marked by an authoritative server or the maximum among all values received from intermediate caches. The hit/miss probabilities derived are approximate in traditional DNS caches receiving different TTLs from higher-level caches and exact in traditional DNS caches getting always their responses from authoritative servers. It is interesting to note that the model of [8] is valid for a single modern DNS cache that overrides the given TTL with a fixed caching duration. Characterizing the traffic not served by the cache (the miss process), considering distributions of caching durations other than the deterministic one, and most challenging extending to the case of a network of caches are issues yet to be addressed.

Bahat and Makowski address in [9] the consistency issue of TTL-based caches in isolation as introduced in [8] (i.e. when timers are set to a fixed constant). They introduce the hit* measure of consistency as the probability to hit the cache and obtain a non-stale data. Closed-form formulas are derived when the original contents are requested through the cache and updated at the server according to two independent renewal point processes. Their results are relevant for a single modern DNS cache that uses deterministic timers. The extension of their single cache model to networks requires the characterization of the miss streams which was not addressed by the authors.

Martina, Garetto and Leonardi propose a unified approach to study systems of caches [10]. They generalize a decoupling technique that allows the study of very large systems of caches at low computational cost. For the single cache case, their approach to study the Random and FIFO cache replacement policy is applicable in the case of a single modern DNS cache. They derive the hit and occupancy probabilities of a single content when requests are Poisson and the caching duration is generally distributed (using an $M/G/1/0$ queue) and when requests form a renewal process and the caching durations are exponentially distributed (using a $G/M/1/0$ queue).

The closest papers to our work, methodologically speaking, are [11, 12, 13, 14]. Choungmo *et al.* analyze both a single cache and a network of caches in which each content remains in cache for a random period [11, 12]. The essential difference with our work is that caching durations are regenerated from the same distribution at each cache hit. As such, the model of [11, 12] applies to modern DNS caches only if caching durations are exponentially distributed, thanks to the memoryless property of the exponential distribution. Observe that the context targeted in [11, 12] is that of ICN architectures. More general TTL-based cache models are studied in [13, 14] where the statistical correlations that request streams may exhibit are accounted for (requests described as Markov renewal processes in [13] and as Markov arrival processes in [14]). Note that Berger *et al.* address three TTL-based caching policies: one that regenerates the TTL upon a hit, another that regenerates the TTL upon a miss and a third policy that combines the two other ones.

It has been reported in [2, 4, 15]—and we have observed it in our collected DNS traces; see Fig. 8—that the sequence of TTLs received relatively to a given resource record exhibits some randomness. We believe it is crucial to consider this randomness when modeling a hierarchy of caches. Another key issue concerns the optimal distribution for the caching durations. Callahan, Allman and Rabinovich mention in [4] that no model or experiment characterizes the optimal (deterministic) TTL choice. We consider a related problem in this paper, namely, when does the deterministic TTL yield optimal performance metrics.

3. Definitions and Assumptions

3.1. Considered Scenario

In this paper, caches are assumed to consist of infinite size buffers. This assumption derives naturally from the fact that the cached entities—the DNS records—have a negligible size when compared to the storage capacity available at a DNS server [8]. The management of different records can then safely be decoupled, simplifying thereby the modeling of caches. Our analysis will focus on a *single* content/record, characterizing the processes relevant to it. The *same* can be *repeated* for every single content requested by users.

Without loss of generality, consider that a *cache miss* occurred at time $m_0 = t_0 = 0$. In other words, the content was not in cache at a request arrival at time t_0. We will neglect the request/record processing time at each server/client and the request/record travel time between servers, as these times are typically

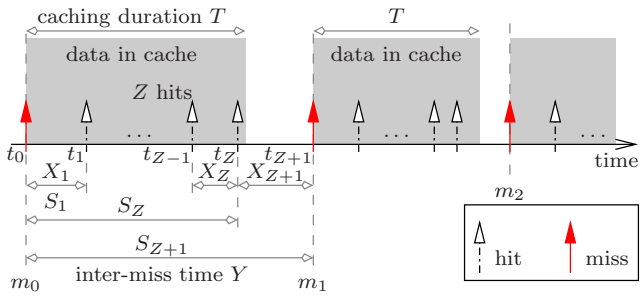


Figure 1: Requests, caching durations and inter-miss times.

insignificant in comparison with the request inter-arrival time. Consequently the content requested is cached and made available to the requester also at time t_0. More precisely, upstream requests and downstream responses are instantaneous.

A cache miss makes the content available in the respective cache for a duration T. Each cache samples this duration from its respective distribution. Caches along the path between the server/client receiving the original request and the server where the content was found all initiate a new duration T at the same time, but the durations initiated being different they will expire at different instants. Consequently, caches become asynchronous, something that would not occur should the caches follow the so-called TTL rule.

Any request arriving during T will find the content in the cache. This is a *cache hit*. The first request arriving after T has expired is a *cache miss* as depicted in Fig. 1 (see for instance instant $t_{Z+1} = m_1$). It initiates a new duration during which the content will be cached.

3.2. Metrics and Properties of a Cache

The performance of a cache policy can be assessed through the computation of several metrics. The *hit probability* h_P captures the chances that a request has to be served by the cache. The *miss probability* m_P is simply the complementary probability. The *hit/miss rate* (h_R/m_R) represents the rate at which cache hits/misses occur. The *occupancy* π is the percentage of time during which the content is cached. We say "a cache policy is *efficient*" if its miss probability is low. This is relevant as long as cached contents are up-to-date.

In fact, by setting timers (or violating the TTL rule in the case of modern DNS), a server/client takes a risk by caching a content for a longer period than it should, as the content may well have changed by the time the locally chosen duration T expires. The cache would then be providing an outdated content. Therefore, it is important to assess the consistency of a cache. Another metric of interest is the cache *refresh rate*). It defines how fast a change in a record can propagate until this cache. High freshness is desirable with dynamic authoritative servers. In practice, it is desirable to have both an efficient and consistent cache, but these are conflicting properties with dynamic servers.

3.3. Processes at Hand

To fully analyze a cache one needs to consider:

The arrival process: it may result from the superposition of multiple independent requests arrival processes. Let $X_k = t_k - t_{k-1}$ be the kth inter-request time ($k > 0$). It is useful to define the kth jump time $S_k = X_1 + X_2 + \ldots + X_k$ with its cumulative distribution function (CDF) $F_{(k)}(t) = \mathbb{P}(S_k < t)$ and its probability density function (PDF) $f_{(k)}(t) = \frac{dF_{(k)}(t)}{dt}$. Let $N(t) = \sup\{k : S_k \leq t\} = \sum_{k>0} \mathbb{1}\{S_k \leq t\}$. The arrival counting process is then $\{N(t), t > 0\}$.

The caching duration: a cache draws the duration T from the same distribution, such that $\mu = 1/\mathbb{E}[T]$. The scenario analyzed here considers memoryless caches, i.e. all caching durations set by the same cache are independent and identically distributed. With a slight abuse of notation, let $T(t)$ be the CDF of the random variable (rv) T.

The outgoing miss process: cache misses form a stochastic process whose inter-miss time is denoted by $Y_k = m_k - m_{k-1}$ for $k > 0$.

The number of hits between consecutive misses: these hits occur within a single caching duration. Their number is a rv denoted by Z.

Observe that, since T is a rv and $N(t)$ the counting variable, $N(T)$ is a rv which represents the number of requests during a caching duration T. As all requests arriving during this period are necessarily hits, following then the above definitions we have that $Z = N(T)$ and its expectation is $\mathbb{E}[Z] = \mathbb{E}[N(T)]$.

In the case of a tree of caches, a subscript referring to the cache label will be added to the rvs for disambiguation. Besides the "instantaneous transmission/processing" assumption that holds throughout this paper, the following holds:

Assumption 1 (Stationary arrivals)**.** *The sequence $\{t_k\}_{k\geq 0}$ forms a stationary point process. And inter-request times $\{X_k\}_{k\geq 1}$ form a stationary and ergodic sequence with finite intensity $\lambda = 1/\mathbb{E}[X_k] < \infty$.*

Assumption 2 (independence)**.** *At any cache, inter-request times and caching durations are independent.*

Assumption 3 (independent arrivals)**.** *Multiple arrivals at any high-level cache are independent.*

Assumption 4 (independent caches)**.** *Caching durations from any two different caches are independent.*

Assumption 1 is general enough to cover most statistical correlations encountered in practical and experimental cases studied in the traffic modelling literature. Assumptions 2 and 4 hold at modern DNS servers [2, 4] and Web browsers [3] as these use their own caching durations independently of the requests and other servers/browsers. Assumption 3 holds if exogenous arrivals are independent, as long as requests for a given content "see" a *polytree* network (that is a directed graph without any undirected cycles).

The example depicted in Fig. 2 shows a general network of caches that is seen by each of the contents as a (poly)tree.

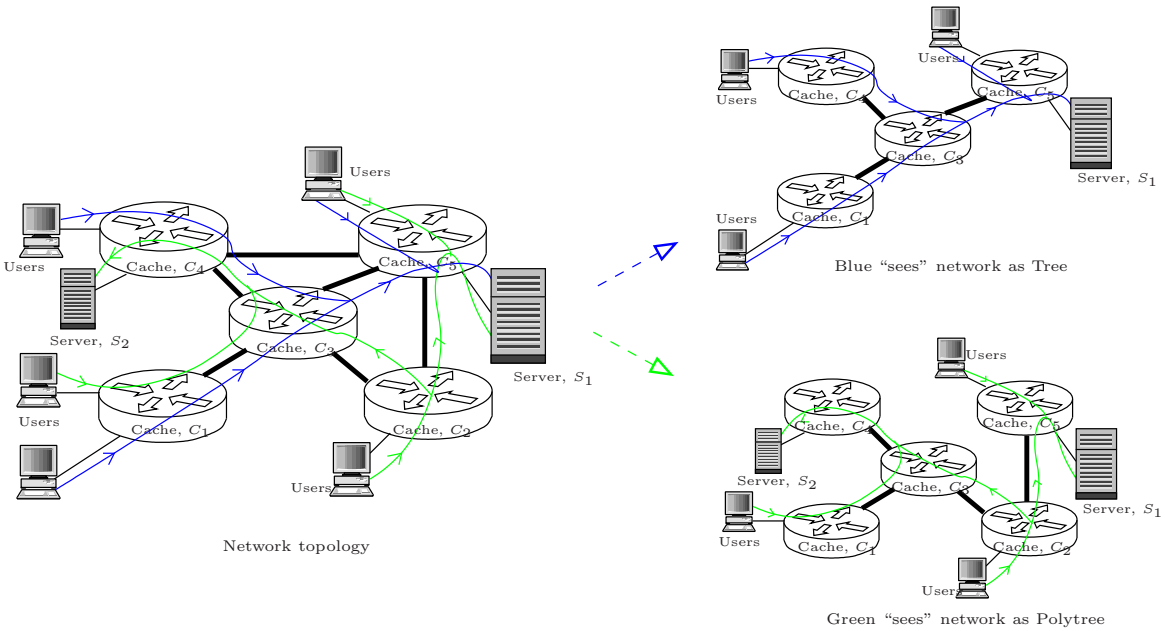


Figure 2: Example of a network where Assumption 3 holds: requests for a given file view the network as a polytree.

It is worth noting that the scenario and the set of assumptions considered here fit the case of a *single traditional* DNS server if the distribution of its caching durations fits the values marking the responses.

Table 1: Glossary of main notation

h_P	hit probability	λ	arrival rate ($1/\mathbb{E}[X]$)
h_R	hit rate	S_k	kth jump time (rv)
m_P	miss probability	$F_{(k)}(t)$	CDF of S_k
m_R	miss rate	$f_{(k)}(t)$	PDF of S_k
π	occupancy	$N(t)$	requests during t (rv)
$\bar{h}_P$	global/average cache hit probability	$M(t)$	renewal function
R_f	refresh rate	$m(t)$	renewal density function
c_P	correctness probability	Z	hits during T (rv)
U	inter-update time (rv)	Y	inter-miss time (rv)
$U(t)$	CDF of U	$G(t)$	CDF of Y
$1/\nu$	expectation of U	$g(t)$	PDF of Y
U_r	forward recurrence update time (rv)	Y_r	interval in r-thinned miss process (rv)
T	caching duration or TTL (rv)	$G_r(t)$	CDF of Y_r
$T(t)$	CDF of T	$g_r(t)$	PDF of Y_r
$1/\mu$	expectation of T	$L(t)$	expected number of hits until t within T
X	inter-request time (rv)	$H(t)$	CDF of inter-request time at higher-level cache
$F(t)$	CDF of X	$\chi^*(s)$	LST of $\chi(t)$
$f(t)$	PDF of X		

Observe also that the popularity of a content is proportional to its request rate λ. Therefore, it should be clear that our models account for a content's popularity (which can be Zipfian, Uniform, Geometric, etc.) through the per-content request rate λ.

A word on the notation: for any function $\chi(t)$, its Laplace-Stieltjes Transform (LST) is $\chi^*(s) = \int_0^\infty e^{-st} d\chi(t)$ $(s \geq 0)$. Observe that the LST of a function is the Laplace transform of its derivative. The complementary cumulative distribution function (CCDF) of a CDF $\chi(t)$ is $\bar{\chi}(t) = 1 - \chi(t)$. Table 1 summarizes the main notation used in the paper.

4. Analysis of a Single Cache

We are ready now to analyze a cache taken in isolation. The results found here will be used in Section 5 when studying multiple caches in a tree network.

4.1. The Model and its Analysis

We start this section by providing closed-form expressions of metrics of interest for a single cache under the general settings described in Section 3.3. Namely, requests are described by stationary point processes which are independent of the caching durations assigned to contents in the cache. The following result provides the hit/miss/occupancy probabilities under this general scope.

Proposition 1. *Under Assumption 1, the miss process of our cache is a stationary point process. Moreover, the hit probability h_P, the miss probability m_P, and the occupancy π are respectively given by*

$$h_P = \frac{\mathbb{E}\left[N(T)\right]}{1 + \mathbb{E}\left[N(T)\right]}, \tag{1}$$

$$m_P = 1 - h_P, \tag{2}$$

$$\pi = \lambda(1 - h_P) \times \mathbb{E}[T], \tag{3}$$

where $\mathbb{E}[.]$ is the expectation with respect to (w.r.t.) the Palm probability of the stationary process $\{N(t), t \geq 0\}$.

Proof. Under Assumption 1 the sequence $\{(t_0, T_0), (t_{Z_1+1}, T_1), \ldots\}$, where the random variables $\{T_i, i \geq 0\}$ are successive realizations of the caching duration T, defines a stationary marked point process [16, Sect. 1.1.3]. The latter stationary point process $\{(t_0, T_0), (t_{Z_1+1}, T_1), \ldots\}$ is the cache miss process. Since the miss process is a stationary process, statistical properties calculated in the first inter-miss time interval with respect to its Palm probability remain valid for the whole process. Hence, the analysis of the cache can be carried out within the first inter-miss time interval.

The derivation of (1) is straightforward from the definition of the hit probability as the ratio of the expected number of requests that hit on the cache (i.e. $\mathbb{E}[Z] = \mathbb{E}[N(T)]$) to the expected total number of requests that arrive on the cache during the first inter-miss time (i.e. $\mathbb{E}[Z] + 1$). Equation (3) is obtained by applying the *Mean-Value Formula* [16, Formula (1.3.2)] which is reproduced here for self-containedness. For any non-negative function g, the time average of $g(Z(t))$, where $\{Z(t), t \in \mathbb{R}\}$ is a stochastic process with measurable values such that $Z(t) = Z(0) \circ \theta_t$ (θ_t can be seen as a shift operator), is given by

$$\text{time average of } g(Z(t)) = \frac{\mathbb{E}\left[\int_0^{T_1} g(Z(t))dt\right]}{\mathbb{E}\big[T_1\big]}. \tag{4}$$

In this formula, $\mathbb{E}$ is the expectation w.r.t. the Palm probability of the stationary process $\{N(t), t \geq 0\}$ and $]0, T_1]$ is the first interval. Thus T_1 must be Y_1, the instant of the first miss. To use (4) for the occupancy probability π, we need to use $Z(t) = \mathbb{1}\{T > t\}$ and $g(z) = \mathbb{1}\{z > 0\}$ so that the time average of $g(Z(t))$ (which is then $\mathbb{1}\{T > t\}$) is π (recall $\pi = P(T > t)$). The integral in (4) becomes

$$\int_0^{Y_1} \mathbb{1}\{T > t\}\, dt \;=\; \int_0^{T} \mathbb{1}\{T > t\}\, dt \;+\; \int_T^{Y_1} \mathbb{1}\{T > t\}\, dt \;=\; T \qquad \Rightarrow \qquad \pi = \frac{\mathbb{E}\big[T\big]}{\mathbb{E}\big[Y_1\big]}.$$

$1/\mathbb{E}[Y_1]$ is nothing but the intensity of the miss process $m_P/\mathbb{E}[X_1]$. Equation (3) is readily found. □

Proposition 1 shows that cache performance metrics are clearly related to (or calculated from) the counting process as long as the request arrival process is a stationary process. The latter might result from the superposition of several sources of requests or the miss streams of other caches in case of a network of caches (see Section 5).

Once performance metrics are computed, our next goal is to characterize the miss process which is the same as the process going out from a cache towards the higher-level server. Assumptions stronger than Assumption 1 on the statistical correlations of the requests streams are needed (e.g. Markov Renewal processes [13] or Markov Arrival processes [14]). We will consider from now on that the following statement holds.

Assumption 5 (renewal arrivals)**.** *Inter-request times $\{X_k\}_{k\geq 1}$ are independent and identically distributed rvs.*

In other words, the request process $\{N(t), t > 0\}$ is a renewal process. Let X be the generic inter-request time, $F(t)$ be its CDF, $f(t) = \frac{dF(t)}{dt}$ be its PDF, and $\lambda = 1/\mathbb{E}[X]$.

Assumption 5 covers a broad range of applications. Point processes can be well approximated by renewal processes [17]. Feldmann and Whitt show in [18] that the long-tailed distributions which are generally observed in network performance analysis can be fitted by a renewal process with a hyper-exponentially distributed interval. Renewal processes with either Weibull (with shape less than 1) or Pareto inter-event distributions are used in [8] by Jung, Berger and Balakrishnan to fit traces collected at DNS servers caches.

The renewal function $M(t)$ and the renewal density function $m(t)$ associated to $\{N(t), t > 0\}$ are, respectively,

$$\begin{aligned} M(t) &= \mathbb{E}[N(t)] = \sum_{k>0} F_{(k)}(t), \\ m(t) &= \tfrac{dM(t)}{dt} = \sum_{k>0} f_{(k)}(t)\ . \end{aligned} \tag{5}$$

It is well-known that the renewal function satisfies the so-called renewal equation [19]

$$M(t) = F(t) + \int_0^t F(t-x)dM(x) = F(t) + \int_0^t F(t-x)m(x)dx. \tag{6}$$

Equivalently, (6) can be written using the PDF $f(t)$

$$M(t) = F(t) + \int_0^t M(t-x)dF(x) = F(t) + (M * f)(t). \tag{7}$$

It will be useful to derive the LST $M^*(s)$ which is also the Laplace transform of $m(t)$. Differentiating (7) yields (use $M(0) = 0$)

$$\begin{aligned} & & m(t) &= f(t) + (m * f)(t) & (8)\\ \text{Laplace transform} \quad &\Rightarrow & M^*(s) &= F^*(s) + M^*(s)F^*(s) & \\ & \Leftrightarrow & M^*(s) &= \frac{F^*(s)}{1 - F^*(s)}. & (9) \end{aligned}$$

Relation (7) between the functions M, F and f is equivalent to Relation (9) between the LSTs M^* and F^*. This equivalency will be used later in Section 5.4.

Remark 1 (Practical considerations)**.** *The renewal function $M(t)$ can be approximated by discretizing the renewal equation [20].*

Proposition 2 (Miss process)**.** *Under Assumptions 2 and 5 the miss process of a single cache is a stationary renewal process.*

Proof. Since the arrival process is a stationary (renewal) process, it follows from the proof of Proposition 1 that the miss process is also a stationary process. Without loss of generality, we assume that the first request arrives at time $t_0 = 0$ while the content is not cached. This cache miss triggers a new caching period. Consequently, miss instants are regeneration points of the state of the cache, implying that these form a renewal process. □

According to Proposition 2 inter-miss times $\{Y_k\}_{k>0}$ are independent and identically distributed. Let Y be the generic inter-miss time and $G(t)$ be its CDF. Computing $G(t)$ will complete the characterization of the miss process. To this end we consider first the number of hits occurring in a renewal interval Y until time t, and more specifically its expectation $L(t)$. We can readily write for $t \geq 0$

$$\begin{aligned} L(t) &= \sum_{k>0} \mathbb{P}(S_k < t, T > S_k) = \sum_{k>0} \int_0^t \mathbb{P}(T > x) dF_{(k)}(x) \\ &= \int_0^t \bar{T}(x) dM(x). \end{aligned} \tag{10}$$

Observe that $\lim_{t\to\infty} L(t)$ is nothing but the expected number of hits in a renewal interval, i.e. $\mathbb{E}[N(T)]$.

Proposition 3 (Inter-miss times)**.** *The CDF $G(t)$ of the generic inter-miss time Y and its LST are given by*

$$G(t) = F(t) - \int_0^t \big(1 - F(t-x)\big) dL(x) \tag{11}$$

$$G^*(s) = 1 - \big(1 - F^*(s)\big)\big(1 + L^*(s)\big). \tag{12}$$

Proof. Let $m_0 = 0$ be the first miss time. The CDF $G(t)$ of the inter-miss time Y can be derived by noticing that $Y = S_{Z+1}$ where Z is the number of hits in a renewal interval ($Z \in \mathbb{N}$). As such, the $(Z+1)$st request occurs after T expires and it will initiate a new renewal interval. By considering the possible values of Z, we can write

$$\begin{aligned} G(t) &= P(S_{Z+1} < t) = \sum_{k\geq 0} P(S_{Z+1} < t, Z = k) \\ &= \sum_{k\geq 0} P(S_k + X_{k+1} < t, S_k < T < S_k + X_{k+1}). \end{aligned}$$

By conditioning first on S_k and then on X_{k+1}, we get

$$\begin{aligned} G(t) &= \sum_{k\geq 0} \int_0^t \left(\int_0^{t-u} (T(u+x) - T(u)) f(x) dx \right) f_{(k)}(u) du \\ &= \sum_{k\geq 0} \int_0^t \left(\int_0^{t-u} T(u+x) f(x) dx \right) f_{(k)}(u) du - \sum_{k\geq 0} \int_0^t T(u) f_{(k)}(u) \left(\int_0^{t-u} f(x) dx \right) du \\ &= \sum_{k\geq 0} \int_0^t \left(\int_u^t T(v) f(v-u) dv \right) f_{(k)}(u) du - \sum_{k\geq 0} \int_0^t T(u) f_{(k)}(u) F(t-u) du \qquad \text{(change } v = u+x\text{)} \\ &= \sum_{k\geq 0} \int_0^t T(v) \left(\int_0^v f(v-u) f_{(k)}(u) du \right) dv - \sum_{k\geq 0} \int_0^t T(u) f_{(k)}(u) F(t-u) du \,. \end{aligned} \tag{13}$$

Equality (13) has been obtained after exchanging the integrals in the first sum. Observe now that, under Assumption 5, the jump time S_k is the sum of k independent rvs that are identically distributed with density f. Hence, The density $f_{(k)}$ of S_k is the k-fold convolution of f, which will be denoted as $f^{(k)}$. We can compute the following

$$\int_0^v f(v-u) f_{(k)}(u) du = (f * f_{(k)})(v) = (f * f^{(k)})(v) = f^{(k+1)}(v) = f_{(k+1)}(v),$$

where we have used the fact that the density of the jump time S_{k+1} is the convolution of $f_{(k)}$ and f, since $S_{k+1} = S_k + X_{k+1}$ and these rvs are independent. We can rewrite (13) as follows

$$\begin{aligned} G(t) &= \sum_{k\geq 0} \int_0^t T(v) f_{(k+1)}(v) dv \; - \sum_{k\geq 0} \int_0^t T(u) f_{(k)}(u) F(t-u) du \\ &= \sum_{k\geq 1} \int_0^t T(x) f_{(k)}(x) dx \; - \sum_{k\geq 0} \int_0^t T(x) f_{(k)}(x) F(t-x) dx \\ &= \sum_{k\geq 1} \int_0^t \big(1 - F(t-x)\big) T(x) f_{(k)}(x) dx \; - \int_0^t T(x) f_{(0)}(x) F(t-x) dx \\ &= \int_0^t \big(1 - F(t-x)\big) T(x) \sum_{k\geq 1} f_{(k)}(x) dx \; - \; 0 \qquad (S_0 = 0 \;\Rightarrow\; f_{(0)}(t) = \delta(0)) \\ &= \int_0^t \big(1 - F(t-x)\big) T(x) dM(x) \qquad \text{(using (5))} \\ &= M(t) - \int_0^t F(t-x) dM(x) - \int_0^t \big(1 - F(t-x)\big) \bar{T}(x) dM(x) \qquad (\text{use } T(x) = 1 - \bar{T}(x)) \\ &= F(t) - \int_0^t \big(1 - F(t-x)\big) \bar{T}(x) dM(x) \end{aligned} \tag{14}$$

where we have used (6) to write (14). The prefinal step is to differentiate (10) which yields $dL(x) = \bar{T}(x)dM(x)$. We can then rewrite (14) as follows

$$G(t) = F(t) - \int_0^t \big(1 - F(t-x)\big)dL(x)$$

which is nothing but (11). To compute $G^*(s)$ we use the fact that the LST of a function is the Laplace transform of its derivative. It is straightforward to rewrite (11) as follows

$$\begin{array}{rrrcll} & & G(t) & = & F(t) - L(t) + (F * L')(t) & \\ \text{differentiating} & \Rightarrow & g(t) & = & f(t) - L'(t) + (f * L')(t) & (\text{use } F(0) = 0) \\ \text{Laplace transform} & \Rightarrow & G^*(s) & = & F^*(s) - L^*(s) + F^*(s)L^*(s) & \\ & & G^*(s) & = & 1 - \big(1 - F^*(s)\big) - L^*(s)\big(1 - F^*(s)\big) & \\ & & G^*(s) & = & 1 - \big(1 - F^*(s)\big)\big(1 + L^*(s)\big) & \end{array}$$

which completes the proof. □

Proposition 3 states that one needs to know the CDF and renewal function of the arrival process and the CDF of the caching duration to derive the CDF of the miss process, or equivalently, the outgoing process. This proposition will be repeatedly used in Section 5 when analyzing networks of caches.

Remark 2. *Although the statement of Proposition 3 is exact, evaluating numerically the CDF of the miss process does not always return exact results. If the renewal function has to be approximated, then the exact analysis may only return approximate numerical results.*

We now focus on the performance metrics defined in Section 3.2. These metrics are for a single content. Similar metrics for the entire set of cache contents can also be defined as long as the contents' popularity is known.

4.1.1. Hit/Miss/Occupancy Probabilities

The following corollary specializes Proposition 1.

Corollary 1 (Cache performance)**.** *Under Assumption 5 the hit probability h_P, the miss probability m_P, and the occupancy π are respectively given by*

$$h_P = \frac{\mathbb{E}[M(T)]}{1 + \mathbb{E}[M(T)]}\ ; \qquad m_P = \frac{1}{1 + \mathbb{E}[M(T)]}\ ; \qquad \pi = \frac{\lambda/\mu}{1 + \mathbb{E}[M(T)]}, \tag{15}$$

where $\mathbb{E}[M(T)] = \int_0^\infty M(x)dT(x)$.

Proof. In the stationary regime, the expected number of hits within a renewal interval is $\mathbb{E}[Z] = \mathbb{E}[N(T)] = \mathbb{E}\left[\mathbb{E}[N(T)|T]\right] = \mathbb{E}[M(T)]$ (where M is the renewal function defined in (6)). Recall that $\lambda = 1/\mathbb{E}[X]$ is the arrival rate and $\mu = 1/\mathbb{E}[T]$ is the caching duration expiration rate. Substituting for $\mathbb{E}[N(T)]$ in (1)-(3) yields (15) and the proof is completed. □

Corollary 1 states that it is enough to compute $\mathbb{E}[M(T)]$ (e.g. $M(T) = \lambda T$ for Poisson processes) and estimate the request rate λ at a cache to derive all its metrics of interest (μ is locally known). It is worth noting that the hit probability h_P and the occupancy π are different in general and in particular under renewal arrival processes. The equality $h_P = \pi$ holds only if the arrival process is a Poisson process thanks to the PASTA (Poisson Arrivals See Time Average) property.

Global cache performance. The global cache performance refers generally to the average hit probability $\bar{h}_P$ over all contents. This metric of interest is calculated as the weighted sum of the file hit probabilities where the weights are the files' popularity. Consider file k, with $k = 1, \ldots, K$; its hit probability is denoted $h_{P,k}$, its requests arrival rate is λ_k and its popularity is $p_k = \lambda_k / \sum_{j=1}^{K} \lambda_j$. The average hit probability is then

$$\bar{h}_P = \sum_{k=1}^{K} p_k h_{P,k}.$$

Note that the content popularity may be Zipfian ($p_k = \kappa \times k^{-\alpha}, \alpha > 0$), Geometric ($p_k = \rho^k, \rho < 1$), etc.

4.1.2. Consistency Metrics

In the case of a dynamic server (i.e. original content located at the server may change over time), it is important to evaluate the consistency of the cache. As the content is cached for a pre-determined duration, requests hitting on the cache may well obtain an outdated content. (TTL-based policies provide weak cache consistency.) For a cache directly connected to the authoritative server, the cache refresh rate R_f is nothing but the miss rate m_R since a content may be refreshed only after T expires. Since λ is the request arrival rate, the cache refresh rate is

$$R_f = \lambda\, m_P = \frac{\lambda}{1 + \mathbb{E}[M(T)]}. \tag{16}$$

When there are intermediate caches between a cache c and the authoritative server, the refresh rate $R_{f,c}$ of cache c is given by the product of miss probabilities at all intermediate caches. Namely,

$$R_{f,c} = \lambda_c \prod_{n \in \mathcal{P}(c)} m_{P,n} \tag{17}$$

where λ_c is the requests arrival rate at cache c, $m_{P,n}$ is the miss probability at cache n, and $\mathcal{P}(c)$ is the set of caches (including cache c) on the path from cache c to the server.

The consistency of a cache is measured in [9] using two metrics, one is the hit* probability and the other is the ratio between the hit* and hit probabilities. The first measures the probability that a cache hit returns a correct content and the second is the ratio of correct hits to all hits which is a way to quantify the quality of the data stored in the cache. Both metrics are derived in the case of a cache directly connected to the server. Similarly to [9], we assume that the content is updated at the server according to an independent renewal point process. We denote by U the generic inter-update time, ν^{-1} its mean, $U(t)$ its CDF, and U_r the stationary forward recurrence update time. The following proposition generalizes Proposition 2 of [9] to the case of generally distributed caching durations.

Proposition 4 (Consistency)**.** *Under Assumptions 2 and 5, and assuming also that server updates are described by an independent renewal process, the hit* probability h_P^* of a cache directly connected to the authoritative server is given by*

$$h_P^* = \frac{\mathbb{E}\big[M(\min\{T, U_r\})\big]}{1 + \mathbb{E}[M(T)\big]} \tag{18}$$

where

$$\mathbb{E}\big[M(\min\{T, U_r\})\big] = \int_0^\infty M(t) dW(t) \tag{19}$$

$$\textit{with} \qquad W(t) = P(\min\{T, U_r\} < t) = T(t) + \frac{1 - T(t)}{\nu} \int_0^t \big(1 - U(u)\big) du\ . \tag{20}$$

Proof. Let Z^* be a rv giving the number of correct hits in a renewal interval. A request at time m_0 will initiate the first inter-miss interval. Hits are those requests arriving in the interval $(m_0, m_0 + T)$ however, only hits occurring in the sub-interval $(m_0, m_0 + \min\{T, U_r(m_0)\})$ will get a correct content. Here $U_r(m_0)$

is the forward recurrence update time at instant m_0. In the stationary regime, the expectation of Z^* is $\mathbb{E}[Z^*] = \mathbb{E}[M(\min\{T, U_r\})]$. Equation (18) is readily found (same denominator as h_P). Let $W(t)$ be the CDF of $\min\{T, U_r\}$. Equation (19) follows immediately. We now compute $W(t)$. We have (use the independence between the caching renewal process and the update process)

$$1 - W(t) = P(T > t, U_r > t) = P(T > t)P(U_r > t) = (1 - T(t))(1 - P(U_r \leq t)) \ .$$

The distribution of the stationary forward recurrence update time is

$$P\big(U_r \leq t\big) = \int_0^t \frac{1 - U(u)}{\nu} du \ .$$

Hence

$$W(t) = T(t) + \frac{1 - T(t)}{\nu} \int_0^t \big(1 - U(u)\big) du$$

and the proof is completed.

□

Observe that (18) is similar to Equation (14) in [9] except that T is deterministic in [9]. The ratio between the hit* and hit probabilities is given by

$$\frac{h_P^*}{h_P} = \frac{\mathbb{E}[Z^*]}{\mathbb{E}[Z]} = \frac{\mathbb{E}[M(\min\{T, U_r\})]}{\mathbb{E}[M(T)]} \ . \tag{21}$$

From a user point of view, what matters is the probability to get a correct content, regardless of whether it came from the cache or from the server. We may then define the *correctness probability*, which we denote by c_P, as the probability to get a correct content. We can write

$$c_P = \frac{1 + \mathbb{E}[Z^*]}{1 + \mathbb{E}[Z]} = \frac{1 + \mathbb{E}[M(\min\{T, U_r\})]}{1 + \mathbb{E}[M(T)]} \ , \tag{22}$$

since the total number of requests answered correctly in a miss renewal period is one (the miss request) plus $\mathbb{E}[Z^*]$ (the correct hits). The ratio h_P^*/h_P used in [9] is a cache consistency measure, whereas c_P can be seen as a measure of quality of service. The extension to the case of intermediate caches is not straightforward and is left for future work.

4.2. Special TTL Distributions

We will consider three particular cases for the distribution of the caching duration and derive the corresponding results.

4.2.1. Deterministic Distribution

We first look at the case when the caching duration is deterministic and equal to the constant D. This setup (single cache, deterministic TTL) is identical to the one in [8].

Result 1 (Deterministic TTL)**.** *The expected number of hits in a renewal interval is* $\mathbb{E}[M(T)] = M(D)$.

Combining Result 1 with Corollary 1 yields the performance metrics. These are exactly the ones found in [8, Thm. 1]. The CDF $G(t)$ of inter-miss times, on the other hand, is a new result. Using $T(t) = \mathbb{1}\{t > D\}$, (11) becomes

$$G(t) = \mathbb{1}\{t > D\} \left(F(t) - \int_0^D (1 - F(t - x)) dM(x) \right) . \tag{23}$$

4.2.2. Exponential Distribution

If caching durations follow an exponential distribution with rate μ, then $T(t) = 1-e^{-\mu t}$ and the following holds.

Result 2 (Exponential TTL)**.** *The expected number of hits in a renewal interval is* $\mathbb{E}[M(T)] = \frac{F^*(\mu)}{1-F^*(\mu)}$, *and* (12) *giving the LST of* $G(t)$ *becomes*

$$G^*(s) = \frac{F^*(s) - F^*(s+\mu)}{1 - F^*(s+\mu)}. \tag{24}$$

The result above is identical to Propositions 3.1 and 3.2 in [11]. The system considered in [11] consists of caches using expiration-based policies whose caching durations are reset at every cache hit. The DNS scenario considered in this paper pre-sets the caching duration at each cache miss. However, when durations are drawn from an exponential distribution, both systems coincide thanks to the memoryless property of the exponential distribution.

4.2.3. Diagonal Matrix-Exponential Distribution

The third particular case considered here is the one of a family of distributions, the so-called *diagonal matrix-exponential* distribution (diag.ME for short). The CDF of an ME distribution can be written as $1 + \boldsymbol{\alpha}\exp(\mathbf{S}t)\mathbf{u}$, where $\boldsymbol{\alpha}$ and $\mathbf{u}$ are dimension-n vectors and $\mathbf{S}$ is an $n \times n$ matrix; the ME distribution is said to have a representation of order n. The class of ME distributions is equivalent to the class of distributions having a rational LST. If $\mathbf{S}$ is diagonal or diagonalizable,[3] then a diag.ME is obtained.

Our interest in the diag.ME is threefold. First, it covers a large set of distributions in particular any mixture of exponentials. Second, as reported in [18], a general point process can be well fitted by a renewal process having a phase-type distribution such as the mixture of exponentials. Third it is analytically tractable as will become clear in Section 5.

The CDF of a caching duration following a diag.ME of order K can be written following Karlin's representation [21]

$$T(t) = 1 - \sum_{k=1}^{K} b_k e^{-\mu_k t}\ , \quad \text{with} \quad \sum_{k=1}^{K} b_k = 1. \tag{25}$$

There is no restrictions on $\{\mu_k\}_{1\le k\le K}$ except that $T(t)$ must be a CDF (observe that $\{b_k\}_{1\le k\le K}$ is not necessarily a probability distribution). The following then holds.

Result 3 (Diag.ME TTL)**.** *The expected caching duration and the expected number of hits are, respectively,*

$$\mu^{-1} = \sum_{k=1}^{K} b_k \mu_k^{-1}; \qquad \mathbb{E}[M(T)] = \sum_{k=1}^{K} \frac{b_k F^*(\mu_k)}{1 - F^*(\mu_k)}, \tag{26}$$

and the LST of $G(t)$ *given in* (12) *can be rewritten*

$$G^*(s) = 1 - \sum_{k=1}^{K} b_k \frac{1 - F^*(s)}{1 - F^*(s+\mu_k)}. \tag{27}$$

Using (26) in Corollary 1 yields the performance metrics.

[3] There exist then an $n \times n$ matrix $\mathbf{P}$ and an $n \times n$ diagonal matrix $\mathbf{A}$ such that $\mathbf{S} = \mathbf{P}\mathbf{A}\mathbf{P}^{-1}$.

4.3. Optimizing the Performance Metrics

This section tackles the following challenging question: which distribution optimizes the performance of a content caching policy and under which conditions? In other words, how can one maximize/minimize the performance metrics?

Caching has been introduced to limit wide-area traffic and to speed up the service to clients. An efficient cache is then one that has a small miss rate or equivalently a high hit probability. In the case of DNS however, a high hit probability translates into a higher probability of serving an outdated content to the users. Indeed, as explained in Section 3, contents are refreshed only upon a cache miss. Having then a minimal miss rate is desirable in a DNS cache when the content is likely to change often. It must be clear from the above that optimizing a DNS caching policy faces conflicting objectives.

In the general case, there is a convex ordering between different caching distributions. Consider two policies: in the first, a content is cached for a deterministic duration D; in the second, the caching duration T has a CDF $T(t)$ such that $\mathbb{E}[T] = \frac{1}{\mu} = D$. The following lemma will be used in the sequel.

Lemma 1. *The following convex ordering holds*

$$D \leq_{\texttt{cx}} T. \tag{28}$$

Proof. The definition of convex ordering of random variables T_1 and T_2 says that $T_1 \leq_{\texttt{cx}} T_2$ if and only if $\mathbb{E}[\phi(T_1)] \leq \mathbb{E}[\phi(T_2)]$ for any convex function ϕ. By Jensen's inequality we know that for any rv T and any convex function ϕ we have

$$\begin{aligned} & \phi(\mathbb{E}[T]) \quad \leq \quad \mathbb{E}[\phi(T)] \\ \Leftrightarrow \qquad & \phi(D) \quad \leq \quad \mathbb{E}[\phi(T)] \ . \end{aligned}$$

This implies the lemma since $\mathbb{E}[\phi(D)] = \phi(D)$. □

Proposition 5 (Properties of deterministic TTL)**.** *When the caching duration is deterministic:*

(i) If the renewal function M of the request process at a cache is concave, then the hit probability is maximized, and both the miss rate and the occupancy are minimized.

(ii) If M is convex, then the hit probability is minimized, and both the miss rate and the occupancy are maximized.

Proof. *(i)* Define $\phi(t) = 1 + M(t)$. We therefore have in Corollary 1

$$h_P(T) = 1 - \frac{1}{\mathbb{E}[\phi(T)]} \ , \qquad m_R(T) = \frac{\lambda}{\mathbb{E}[\phi(T)]} \ , \qquad \pi(T) = \frac{\lambda D}{\mathbb{E}[\phi(T)]} \ .$$

We have appended to the notation the rv T to stress that the performance metrics depend on the distribution of T. If M is concave, then ϕ is also concave. Since $D \leq_{\texttt{cx}} T$ (Lemma 1), the following holds

$$\mathbb{E}[\phi(D)] \geq \mathbb{E}[\phi(T)] \qquad \Rightarrow \qquad \begin{cases} h_P(D) = 1 - \frac{1}{\mathbb{E}[\phi(D)]} \ \geq \ 1 - \frac{1}{\mathbb{E}[\phi(T)]} = h_P(T) \\ m_R(D) = \frac{\lambda}{\mathbb{E}[\phi(D)]} \ \leq \ \frac{\lambda}{\mathbb{E}[\phi(T)]} = m_R(T) \\ \pi(D) = \frac{\lambda D}{\mathbb{E}[\phi(D)]} \ \leq \ \frac{\lambda D}{\mathbb{E}[\phi(T)]} = \pi(T) \ . \end{cases}$$

(ii) If M is convex then

$$\mathbb{E}[\phi(D)] \leq \mathbb{E}[\phi(T)] \qquad \Rightarrow \qquad \begin{cases} h_P(D) = 1 - \frac{1}{\mathbb{E}[\phi(D)]} \ \leq \ 1 - \frac{1}{\mathbb{E}[\phi(T)]} = h_P(T) \\ m_R(D) = \frac{\lambda}{\mathbb{E}[\phi(D)]} \ \geq \ \frac{\lambda}{\mathbb{E}[\phi(T)]} = m_R(T) \\ \pi(D) = \frac{\lambda D}{\mathbb{E}[\phi(D)]} \ \geq \ \frac{\lambda D}{\mathbb{E}[\phi(T)]} = \pi(T) \ . \end{cases}$$

The proof is complete. □

Proposition 5(i) is applicable for instance when the inter-request time has a decreasing failure rate, as Brown has shown in [22, Theorem 3] that this is a sufficient condition for $M(t)$ to be concave. (The distribution of X has a decreasing failure rate if $P(X > t + s)/P(X > t)$ is increasing in t for each $s > 0$.) Observe that mixing exponential distributions results in a distribution with a decreasing failure rate [22]. Note also that both the Pareto and the Weibull distribution (with shape less than one) have a decreasing failure rate. Such distributions have been used in [8] to fit collected inter-request times at DNS caches (see discussion around Assumption 5).

Proposition 6 (Insensitivity of metrics to TTL distribution)**.** *The miss rate, the hit probability and the occupancy are insensitive to the distribution of the caching duration if the renewal function M of the request process at a cache is linear.*

Proof. Define $\phi(t) = 1 + M(t)$. If M is linear then $\mathbb{E}[\phi(T)] = \phi(\mathbb{E}[T]) = \phi(D) = 1 + M(D)$. For any CDF $T(t)$, the performance metrics are

$$h_P = 1 - \frac{1}{1 + M(D)]} , \qquad m_R = \frac{\lambda}{1 + M(D)} , \qquad \pi = \frac{\lambda D}{1 + M(D)} . \tag{29}$$

The performance metrics depend only on the expectation $\mathbb{E}[T] = D$ and are insensitive to the distribution of T. □

Proposition 6 is applicable for instance when the request process is Poisson since we would have $M(t) = \lambda t$. Observe that the so-called *independent reference model* (IRM) that is often used in the analysis of caches (e.g. [23, 24, 25]) is equivalent to assuming that requests for a single content form a Poisson process [26].

4.4. Applicability to a Traditional DNS Cache

The modern DNS cache analyzed in Section 4 holds the content for a locally chosen duration. Instead, in a traditional DNS cache, the caching duration is the one advocated by the answerer. What matters in the analysis of a single cache is the distribution of the caching durations and not whether the distribution is set locally or it is imposed. Therefore, the findings of Section 4 apply in the case of a single traditional DNS cache, as long as Assumptions 2–5 hold. Note that Jung *et al.* consider in [8] a deterministic caching duration set to the maximum value among all those observed in the responses. If the cache is directly connected to the authoritative server, then the model developed in [8] will be exact. In the cases of intermediate caches, the answerer which is not an authoritative server will provide responses having varied TTL values. Consequently, the model in [8] provides *approximate* results for a single traditional DNS cache *every time the answerer is not an authoritative server*. Instead our model yields *exact* results for both a traditional cache *and* a modern cache, regardless of the distribution chosen for the whole range of caching durations.

5. Analysis of a Cache Network

Section 4 focused on results for a single cache. In this section, we will extend these results for the case where we have caches at multiple nodes (e.g. client, ADSL modem, Internet server provider's DNS server, authoritative server). We say that we have a *network of caches*.

Throughout this section, Assumptions 4-5 are enforced and exogenous arrivals are assumed to be independent. Requests for a given content may only flow over a (poly)tree network so that Assumption 3 holds. The notation relative to cache c will have an extra subscript "c".

To analyze a network of cache, one additionally needs to consider the network topology. In the following we consider the particular case of linear networks for which exact analytical results can be derived (see Section 5.1). Exact partial results can be derived in star networks and in linear star networks (see Section 5.2). We will move next to the general tree network case for which approximate results can be derived (see Section 5.3). The case of polytree networks is addressed in Section 5.4. Last, we focus on the particular case where caching durations and exogenous inter-request times follow a diag.ME distribution (see Section 5.5). Results for this last case are interesting as the diag.ME distribution will be preserved inside the network.

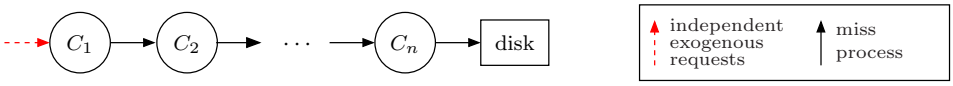
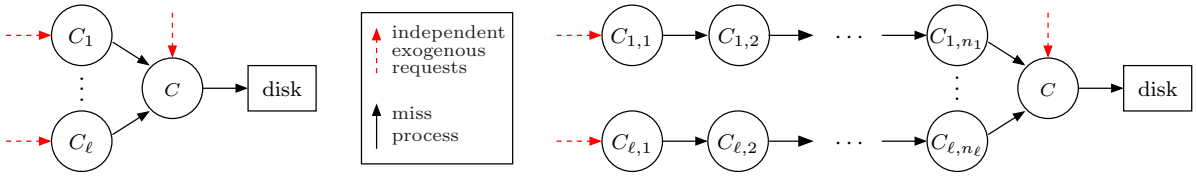


Figure 3: A line of caches.

5.1. Linear Networks: Exact Analytical Results

Consider the line of caches depicted in Fig. 3. There are n caches in a line and the disk of the authoritative server (the rightmost cache is the one of the authoritative server). By Assumption 5, the request process at cache C_1 is a renewal process. By Proposition 3, the miss process at cache C_1 is also a renewal process. This miss process is nothing but the request process at cache C_2. By the same proposition, the miss process at cache C_2 is again a renewal process, so on and so forth. Thus, all processes in this linear network of caches $\{C_1, \ldots, C_n\}$ are renewal processes. The distribution of the miss process and the performance metrics at each cache are derived using Proposition 3 and Corollary 1, respectively.

5.2. (Linear) Star Networks: Exact Partial Results



(a) a star network

(b) a linear star network

Figure 4: Networks where performance metrics can be derived.

We now consider the "star" network depicted in Fig. 4a. This is a one-level tree network composed of ℓ leaves and one root. Under Assumption 5, the requests arrival process at cache C_i ($i = 1, \ldots, \ell$) is a renewal process, then by Proposition 3 the miss process at cache C_i is also a renewal process. As all exogenous processes are independent, all miss processes of caches $C_1, \ldots, C_\ell$ are independent and further independent of the exogenous request process at cache C. As a result, the root cache C is fed by the superposition of $\ell+1$ independent renewal processes which is a stationary ergodic process. The conditions to use Proposition 1 are satisfied and the hit and occupancy probabilities can be computed accordingly.

The linear star network shown in Fig. 4b is the composition of a star network and ℓ linear networks. The results of Section 5.1 are applicable within each line of caches $\{C_{i,1}, \ldots, C_{i,n_i}\}$, for $i = 1, \ldots, \ell$. The performance metrics and the distribution of the miss process can be computed at any cache $C_{i,j}$, for $j = 1, \ldots, n_i$ and $i = 1, \ldots, \ell$ using Corollary 1 and Proposition 3, respectively. Thus, each of the processes arriving to the root cache C is a renewal process and their superposition is a stationary ergodic process. The hit and occupancy probabilities follow from Proposition 1.

5.3. Tree Networks: Multiple Sources of Requests

The exact analysis in Section 5.1 cannot be easily extended to general networks. In fact, in the presence of exogenous requests or several independent request streams, the aggregate (overall) arrival process at a cache is in general not a renewal process (it would be if requests are Poisson for instance). Hence, we cannot apply Proposition 3 that allows us to characterize the distribution of the miss process at the higher-layer cache. To overtake this limitation, we proceed as in [11]: we approximate the aggregate process with a renewal process and assess the quality of this approximation through a numerical study in Section 6.2. The solution derived produces highly accurate approximations for all metrics computed using Corollary 1 (i.e. hit/miss probabilities, hit/miss rates, cache occupancy). This approximation is based on the following statement:

Approximation 1 (aggregation)**.** *The overall request arrival process at each cache is a renewal process.*

Note that for leaf caches, Approximation 1 is simply Assumption 5. Note also that for Poisson requests the statement of Approximation 1 is true. It is worth mentioning that under certain conditions (the asymptotic sum of intervals' CDFs grows linearly with time) the superposition of many sparse renewal processes tends asymptotically to a Poisson process [27, Sect. 5.9]. In cache networks, the request process gets sparser when moving upstream. The result of [27, Sect. 5.9] suggests that the larger the network is, the better the approximation will be. In the case of DNS, the hierarchy is very flat, suggesting that Approximation 1 would be tight.

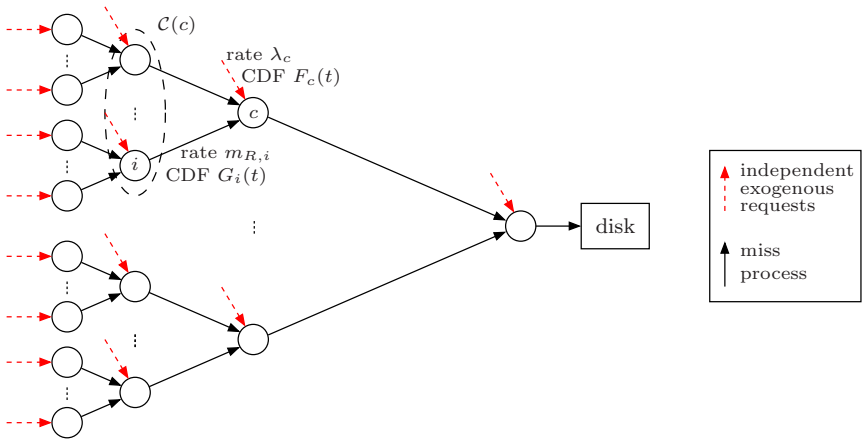


Figure 5: A tree network. Some of the notation relative to cache c are reported.

Consider the tree network depicted in Figure 5 and focus on a given cache c. The *exogenous* requests process (if any) at cache c has rate λ_c and the CDF of its inter-request time is $F_c(t)$. The set of children of cache c is $\mathcal{C}(c)$ with $|\mathcal{C}(c)| = C$. The miss process at child i, with rate $m_{R,i}$ and CDF of inter-miss time $G_i(t)$, is seen as one of the requests processes entering cache c.

We now characterize the *aggregate* request process at cache c. It is composed of the superposition of the C miss processes at the children of c and the exogenous request process at cache c. The rate of the aggregate request process is

$$\Lambda_c = \lambda_c + \sum_{i \in \mathcal{C}(c)} m_{R,i}. \tag{30}$$

Since all exogenous processes are independent, the $C+1$ request processes at cache c are also independent. Thereby, the result derived by Lawrance in [28, Eq. (4.1)] applies. By Approximation 1, the aggregate request process at cache c is a renewal process. Let $H_c(t)$ be the CDF of its inter-request time and $M_c(t)$ the renewal function associated with it. We can write

$$\bar{H}_c(t) = \frac{\lambda_c}{\Lambda_c}\bar{F}_c(t) \prod_{i \in \mathcal{C}(c)} m_{R,i} \int_t^\infty \bar{G}_i(u)du + \sum_{i \in \mathcal{C}(c)} \frac{m_{R,i}}{\Lambda_c}\bar{G}_i(t)\lambda_c \int_t^\infty \bar{F}_c(u)du \prod_{\substack{j \in \mathcal{C}(c) \\ j \neq i}} m_{R,j} \int_t^\infty \bar{G}_j(u)du. \tag{31}$$

We now move to the characterization of the miss process at cache c. A direct consequence of Approximation 1 is that the miss process at each cache is a renewal process thanks to Proposition 2. Proposition 3 and Corollary 1 are also valid at any cache.

Recall that in the case of a single cache, the CDF $G(t)$ of the inter-miss time at a cache is expressed as a function of $F(t)$, the CDF of the inter-request time, and of $M(t)$, the renewal function of the request process; see (14). In the case of a tree network, one needs to consider the *aggregate* request process at cache c. Replacing $F(t)$ with $H_c(t)$ and $M(t)$ with $M_c(t)$ in (14) yields $G_c(t)$, the CDF of the inter-miss time at cache c. More precisely, (14) becomes

$$G_c(t) = H_c(t) - \int_0^t (1 - H_c(t-x))\bar{T}_c(x)dM_c(x) \tag{32}$$

where $\bar{T}_c(t)$ is the CCDF of the caching duration at cache c.

Equations (31) and (32) provide a recursive procedure for calculating the CDFs $H_c(t)$ and $G_c(t)$ at each cache c of a tree network, starting from the lower levels of the tree and moving upward. Numerical procedures such as Romberg's method or other techniques for computing (31) and (32) recursively can be found in [20]. In a special case presented in Section 5.5 $H_c(t)$ and $G_c(t)$ will be found in closed-form.

5.4. Polytree Networks: Multiple Destinations

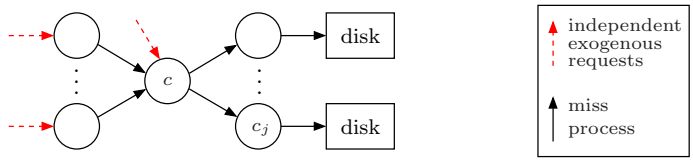


Figure 6: A polytree network.

We consider now the case of polytrees as illustrated in Figure 6. Missed requests at a cache c may be forwarded to more than one higher layer cache. In the DNS system this occurs when multiple authoritative servers handle the same record for fault tolerance, load balancing, etc. The results found in Section 5.3 can be generalized by considering thinned miss processes instead of miss processes.

Consider cache c in Figure 6 and let r be the probability that a miss request at cache c is forwarded to cache c_j. (More generally, we may consider that the probability to choose the outgoing link j is r_j such that $\sum_j r_j = 1$.) The resulting request process arriving to cache c_j and coming from cache c is called the *r-thinned miss process* of cache c.

Under Approximation 1, the miss process at cache c is a renewal process. Therefore, its r-thinned process is also a renewal process [29]. To characterize the distribution of the r-thinned miss process, we introduce the rv W as the number of successive miss requests forwarded to other than cache c_j, we have $P(W = i) = (1 - r)^i r$. The probability density function $g_{c,r}(t)$ of the inter-request times of the r-thinned miss process is the expectation of the $(W + 1)$-fold convolution of the probability density function $g_c(t)$ of the inter-request times of the original miss process. More precisely,

$$g_{c,r}(t) = \mathbb{E}\left[g_c^{(W+1)(t)}\right] = \sum_{i=0}^{\infty} g_c^{(i+1)}(t)(1-r)^i r,$$

where $g_c^{(n)}$ is the n-fold convolution of g_c. By taking the Laplace transform, the LST of the CDF $G_{c,r}(t)$ of the r-thinned miss process is obtained as a function of $G_c^*(s)$ as follows

$$G_{c,r}^*(s) = \frac{rG_c^*(s)}{1-(1-r)G_c^*(s)}. \tag{33}$$

We have a relation between the LSTs $G_{c,r}^*$ and G_c^*. An equivalent relation can be written between the functions $G_{c,r}$, G_c and g_c, using the equivalency found in Section 4.1. The following holds

$$G_{c,r}(t) \;=\; rG_c(t) + (1-r)(G_{c,r} * g_c)(t) \;=\; rG_c(t) + (1-r)\int_0^t g_c(t-x)G_{c,r}(x)dx. \tag{34}$$

This equation is similar in structure to the renewal equation.

Remark 3 (Feed-Forward Networks)**.** *The analysis of feed-forward networks (where a request cannot visit a cache twice) is derived from that of polytrees by relaxing Assumption 3. Therefore, request streams arriving at a cache may not be independent and the characterization of the overall request process using Lawrance's formula [28, Eq. (4.1)] may not be accurate.*

5.5. Closed-Form Results with diag.ME RVs

In this section, we consider a polytree network where caching durations at any cache follow a diag.ME distribution. Also, we will consider that the *exogenous* request process at any cache is a renewal process whose inter-request time follows a diag.ME distribution. More precisely, at a cache c we have for $t > 0$

$$F_c(t) = 1 - \sum_{j=1}^{J_c} a_{c,j} e^{-\lambda_{c,j} t} \ , \qquad \text{with} \quad \sum_{j=1}^{J_c} a_{c,j} = 1 \ , \tag{35}$$

$$\bar{T}_c(t) = \sum_{k=1}^{K_c} b_{c,k} e^{-\mu_{c,k} t} \ , \qquad \text{with} \quad \sum_{k=1}^{K_c} b_{c,k} = 1 \ . \tag{36}$$

J_c and K_c are the respective orders of the diag.ME distributions. We will first prove interesting properties of the diag.ME distribution before stating the main result of this section.

Proposition 7 (Miss process with diag.ME)**.** *If inter-request times and caching durations are diag.ME distributed, then the miss process is a diag.ME renewal process with known representation.*

Proof. We know from Proposition 2 that the miss process is a renewal process and the CDF of the inter-miss time (in the general case) is given in Proposition 3. When the CDF of the inter-request time at a cache c is given by (35), the renewal equation (7) becomes

$$M_c(t) \ = \ F_c(t) + \int_0^t f_c(t-x) M_c(x) dx \ = \ F_c(t) + \int_0^t \sum_{j=1}^{J_c} a_{c,j} \lambda_{c,j} e^{-\lambda_{c,j}(t-x)} M_c(x) dx. \tag{37}$$

The solution of (37) is found in [30, Sect. 2.2.1.19], namely

$$M_c(t) \ = \ F_c(t) + \int_0^t \sum_{j=1}^{J_c} \gamma_{c,j} e^{-\theta_{c,j}(t-x)} F_c(x) dx \tag{38}$$

where $(\theta_{c,j})_{1 \le j \le J_c}$ are the J_c roots of the algebraic equation

$$1 - \sum_{j=1}^{J_c} \frac{a_{c,j} \lambda_{c,j}}{\lambda_{c,j} - z} \ = \ 0, \tag{39}$$

and $(\gamma_{c,j})_{1 \le j \le J_c}$ are the solution of the linear system of J_c equations

$$\left\{ 1 - \sum_{j=1}^{J_c} \frac{\gamma_{c,j}}{\lambda_{c,n} - \theta_{c,j}} \ = \ 0, \qquad 1 \le n \le J_c. \right. \tag{40}$$

Differentiating (38) yields (use $F_c(0) = 0$)

$$M_c'(t) \ = \ F_c'(t) + \int_0^t \sum_{j=1}^{J_c} \gamma_{c,j} e^{-\theta_{c,j}(t-x)} F_c'(x) dx$$

$$\text{Laplace transform} \quad \Rightarrow \quad M_c^*(s) \ = \ F_c^*(s) + \sum_{j=1}^{J_c} \frac{\gamma_{c,j}}{s + \theta_{c,j}} F_c^*(s) \ . \tag{41}$$

Solving for $M_c^*(s)$ in the system of equations composed of (9) and (41) yields

$$M_c^*(s) \ = \ \sum_{j=1}^{J_c} \frac{\gamma_{c,j}}{s + \theta_{c,j}} \qquad \Rightarrow \qquad dM_c(t) \ = \ \sum_{j=1}^{J_c} \gamma_{c,j} e^{-\theta_{c,j} t} dt \ . \tag{42}$$

When the CDF of caching durations at a cache c is given by (36), we can rewrite (14) as follows (use (35) and (42))

$$\begin{aligned} G_c(t) &= 1 - \sum_{j=1}^{J_c} a_{c,j} \left(1 - \sum_{k=1}^{K_c}\sum_{i=1}^{J_c} \frac{b_{c,k}\gamma_{c,i}}{\lambda_{c,j} - \theta_{c,i} - \mu_{c,k}}\right) e^{-\lambda_{c,j}t} \\ &\quad - \sum_{k=1}^{K_c}\sum_{i=1}^{J_c} \left(\sum_{j=1}^{J_c} \frac{a_{c,j}b_{c,k}\gamma_i}{\lambda_{c,j} - \theta_{c,i} - \mu_{c,k}}\right) e^{-(\theta_{c,i}+\mu_{c,k})t}. \end{aligned} \tag{43}$$

Clearly, the inter-miss time at cache c follows a diag.ME distribution. This Karlin representation has order $J_c(K_c+1)$ but it is not guaranteed to be the minimal order. The proof is complete. □

Proposition 8 (Closure of diag.ME under thinning)**.** *The thinning of a renewal process with a diag.ME distributed interval is a renewal process with a diag.ME distributed interval.*

Proof. Consider a renewal process whose interval has the following CDF

$$F(t) = 1 - \sum_{j=1}^{J} a_j e^{-\lambda_j t}\ , \qquad \text{with} \quad \sum_{j=1}^{J} a_j = 1\ .$$

According to (34), the CDF of the interval of the r-thinned version of this renewal process is

$$\begin{aligned} F_r(t) &= rF(t) + (1-r)\int_0^t F'(t-x)F_r(x)dx \\ &= rF(t) + \int_0^t \sum_{j=1}^{J}(1-r)a_j\lambda_j e^{-\lambda_j(t-x)}F_r(x)dx\ . \end{aligned} \tag{44}$$

Equation (44) is similar to (37); its solution is

$$F_r(t) \ = \ r\left(F(t) + \int_0^t \sum_{j=1}^{J} \gamma_j e^{-\theta_j(t-x)}F(x)dx\right) \tag{45}$$

where $(\theta_j)_{1\le j\le J}$ are the J roots of the algebraic equation

$$1 - (1-r)\sum_{j=1}^{J}\frac{a_j\lambda_j}{\lambda_j - z} \ = \ 0, \tag{46}$$

and $(\gamma_j)_{1\le j\le J}$ are the solution of the linear system of J equations

$$\left\{1 - \sum_{j=1}^{J}\frac{\gamma_j}{\lambda_n - \theta_j} \ = \ 0, \qquad 1 \le n \le J. \right. \tag{47}$$

Starting from (45), we can compute

$$\begin{aligned} F_r(t) &= r\left(1 - \sum_{i=1}^{J} a_i e^{-\lambda_i t} + \sum_{j=1}^{J}\gamma_j e^{-\theta_j t}\int_0^t e^{\theta_j x}dx - \sum_{j=1}^{J}\gamma_j e^{-\theta_j t}\sum_{i=1}^{J} a_i \int_0^t e^{-(\lambda_i-\theta_j)x}dx\right) \\ &= r\left(1 - \sum_{i=1}^{J} a_i e^{-\lambda_i t} + \sum_{j=1}^{J}\gamma_j\frac{1-e^{-\theta_j t}}{\theta_j} + \sum_{j=1}^{J}\gamma_j\sum_{i=1}^{J} a_i\frac{e^{-\lambda_i t}}{\lambda_i - \theta_j} - \sum_{j=1}^{J}\gamma_j e^{-\theta_j t}\sum_{i=1}^{J}\frac{a_i}{\lambda_i - \theta_j}\right) \\ &= r\left(1 + \sum_{j=1}^{J}\frac{\gamma_j}{\theta_j} - \sum_{i=1}^{J} a_i\left(1 - \sum_{j=1}^{J}\frac{\gamma_j}{\lambda_i - \theta_j}\right)e^{-\lambda_i t} - \sum_{j=1}^{J}\gamma_j\left(\frac{1}{\theta_j} + \sum_{i=1}^{J}\frac{a_i}{\lambda_i - \theta_j}\right)e^{-\theta_j t}\right)\ . \end{aligned} \tag{48}$$

The coefficient of $e^{-\lambda_i t}$ is null due to (47). The coefficient of $e^{-\theta_j t}$ can be rewritten

$$\frac{1}{\theta_j}+\sum_{i=1}^{J}\frac{a_i}{\lambda_i-\theta_j} \;=\; \frac{1}{\theta_j}\left(\sum_{i=1}^{J}a_i\right)+\sum_{i=1}^{J}\frac{a_i}{\lambda_i-\theta_j} \;=\; \sum_{i=1}^{J}a_i\left(\frac{1}{\theta_j}+\frac{1}{\lambda_i-\theta_j}\right) \;=\; \frac{1}{\theta_j}\sum_{i=1}^{J}\frac{a_i\lambda_i}{\lambda_i-\theta_j} \;=\; \frac{1}{\theta_j}\frac{1}{1-r}$$

where the last equality is due to (46). Equation (48) simplifies to

$$F_r(t) \;=\; r\left(1+\sum_{j=1}^{J}\frac{\gamma_j}{\theta_j}\right)-\frac{r}{1-r}\sum_{j=1}^{J}\frac{\gamma_j}{\theta_j}e^{-\theta_j t} \;=\; 1-\frac{r}{1-r}\sum_{j=1}^{J}\frac{\gamma_j}{\theta_j}e^{-\theta_j t} \tag{49}$$

where we used $\lim_{t\to\infty}F_r(t)=1$. The CDF in (49) is a diag.ME with order J. The proof is complete. □

Proposition 9 (Closure of diag.ME under aggregation)**.** *The aggregation of independent processes each with a diag.ME distributed interval is a renewal process with a diag.ME distributed interval.*

Proof. Without loss of generality the proof is conducted for the aggregation of $C+1$ processes. By Assumption 5 and Approximation 1, all processes at hand are renewal processes. We focus then on the distribution of the inter-event time of the aggregate process. Let F_i be the CDF of the interval of the ith process. We have for $t>0$ and $i=0,1,\ldots,C$

$$F_i(t)=1-\sum_{l_i=1}^{\mathcal{L}_i}a_{i,l_i}e^{-\lambda_{i,l_i}t}\;,\qquad \text{with}\quad \sum_{l_i=1}^{\mathcal{L}_i}a_{i,l_i}=1\qquad (\text{diag.ME distribution of order }\mathcal{L}_i).$$

The rate of process i is denoted $r_i=\sum_{l_i=1}^{\mathcal{L}_i}a_{i,l_i}\lambda_{i,l_i}$. The rate of the aggregate process is $\Lambda=\sum_{i=0}^{C}r_i$. Given that all processes are independent, the result of Lawrance in [28, Eq. (4.1)] can be used. The CCDF $\bar{H}(t)$ of the interval of the aggregate process is then

$$\begin{aligned}
\bar{H}(t) &= \sum_{i=0}^{C}\frac{r_i}{\Lambda}\bar{F}_i(t)\prod_{\substack{j=0,\ldots,C\\ j\neq i}}r_j\int_t^\infty\bar{F}_j(u)du\\
&= \sum_{i=0}^{C}\frac{r_i}{\Lambda}\sum_{l_i=1}^{\mathcal{L}_i}a_{i,l_i}e^{-\lambda_{i,l_i}t}\prod_{\substack{j=0,\ldots,C\\ j\neq i}}r_j\int_t^\infty\sum_{l_j=1}^{\mathcal{L}_j}a_{j,l_j}e^{-\lambda_{j,l_j}u}du\\
&= \sum_{i=0}^{C}\frac{r_i}{\Lambda}\sum_{l_i=1}^{\mathcal{L}_i}a_{i,l_i}e^{-\lambda_{i,l_i}t}\prod_{\substack{j=0,\ldots,C\\ j\neq i}}r_j\sum_{l_j=1}^{\mathcal{L}_j}\frac{a_{j,l_j}}{\lambda_{j,l_j}}e^{-\lambda_{j,l_j}t}\\
&= \frac{\prod_{i=0}^{C}r_i}{\Lambda}\sum_{i=0}^{C}\sum_{l_i=1}^{\mathcal{L}_i}a_{i,l_i}e^{-\lambda_{i,l_i}t}\prod_{\substack{j=0,\ldots,C\\ j\neq i}}\sum_{l_j=1}^{\mathcal{L}_j}\frac{a_{j,l_j}}{\lambda_{j,l_j}}e^{-\lambda_{j,l_j}t}.
\end{aligned}\tag{50}$$

After tedious calculations, (50) can be rewritten

$$\bar{H}(t) \;=\; \frac{\prod_{i=0}^{C}r_i}{\Lambda}\sum_{l_0=1}^{\mathcal{L}_0}\sum_{l_1=1}^{\mathcal{L}_1}\cdots\sum_{l_C=1}^{\mathcal{L}_C}\left(\sum_{i=0}^{C}\lambda_{i,l_i}\right)\left(\prod_{j=0}^{C}\frac{a_{j,l_j}}{\lambda_{j,l_j}}\right)\exp\left(-\left[\sum_{j=0}^{C}\lambda_{j,l_j}\right]t\right). \tag{51}$$

The interval of the aggregate process follows a diag.ME distribution. Note that the order of this representation is $\prod_{i=0}^{C}\mathcal{L}_i$ but it is not guaranteed that this representation is minimal. The proof is complete. □

We are now in position to write the main result of this section that is the self-preservation of the diag.ME distribution across a polytree network as stated in what follows.

Proposition 10 (Diag.ME closure on polytrees)**.** *Under Assumption 5 and Approximation 1 and as long as* (35)-(36) *are verified at each cache c of a polytree network, miss processes, thinned miss streams, and aggregate requests are all renewal processes with diag.ME distributed intervals.*

Proof. The proof is readily found using iteratively Propositions 7-9 starting at the leaves of the polytree and then moving upwards. The miss process at each lowest level cache checks Proposition 10 thanks to Proposition 7. At any higher level cache, (*i*) Proposition 8 ensures that thinned requests streams from lower caches check Proposition 10, (*ii*) Proposition 9 ensures that the aggregation of these requests streams checks Proposition 10, and (*iii*) Proposition 7 ensures the cache miss process checks Proposition 10. □

The performance metrics can be found at each cache by using Result 3 and Corollary 1. It is important to start the computation with the lowest-level caches as their miss rates will be used to derive $H_c(t)$ at a higher-level cache. It is also $H_c^*(s)$ that should be used instead of $F^*(s)$ in Result 3 at each higher-level cache.

Sections 5.3-5.5 provide approximate results since Approximation 1 is used. The robustness of our model is tested in Section 6.2.

6. Validation, Numerical Results

The objective of this section is to test the robustness of our models against violations of the main assumptions. We first address the case of a single cache by comparing the analytic results of Section 4 to results derived from a real DNS cache trace. The case of a network of caches is addressed next.

6.1. Single Cache Case: Using a Real Trace

In this section, we use traces collected from a real DNS cache to assess the robustness of our analysis. Our home institution Inria at Sophia Antipolis manages two DNS servers in parallel to ensure a good load balancing. The DNS traffic at one of these servers has been collected from 21 June to 1 July 2013 (a duration of 9 days, 23 hours, 19 minutes and 58 seconds). The trace contains information about 2 599 607 resource records requested a total of 15 376 226 times by a total of 889 users. A pre-processing of this trace provided for 2 258 086 resource records (or contents) the following information:

1. the requests instants (from users to Inria's DNS server);
2. the cache miss instants (coinciding with the instants of requests from Inria's DNS server to Internet);
3. the responses instants (from Internet to Inria's DNS server);
4. the final responses instants (from Inria's DNS server to users);
5. the TTL values (in response packets).

Having this information for each content allows us to compute the popularity distribution. A simple fit reveals that the popularity in the arrival process follows Zipf law of parameter $\alpha \approx 1$ as shown in Fig. 7. This is also consistent with the well-known characteristics of Internet traffic.

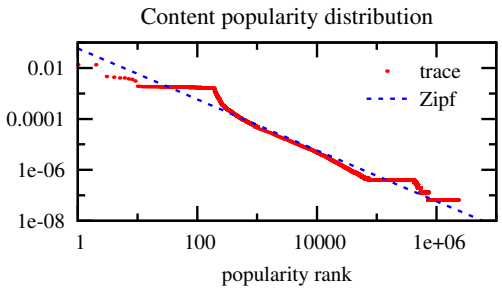


Figure 7: Popularity of resource records at Inria's DNS server.

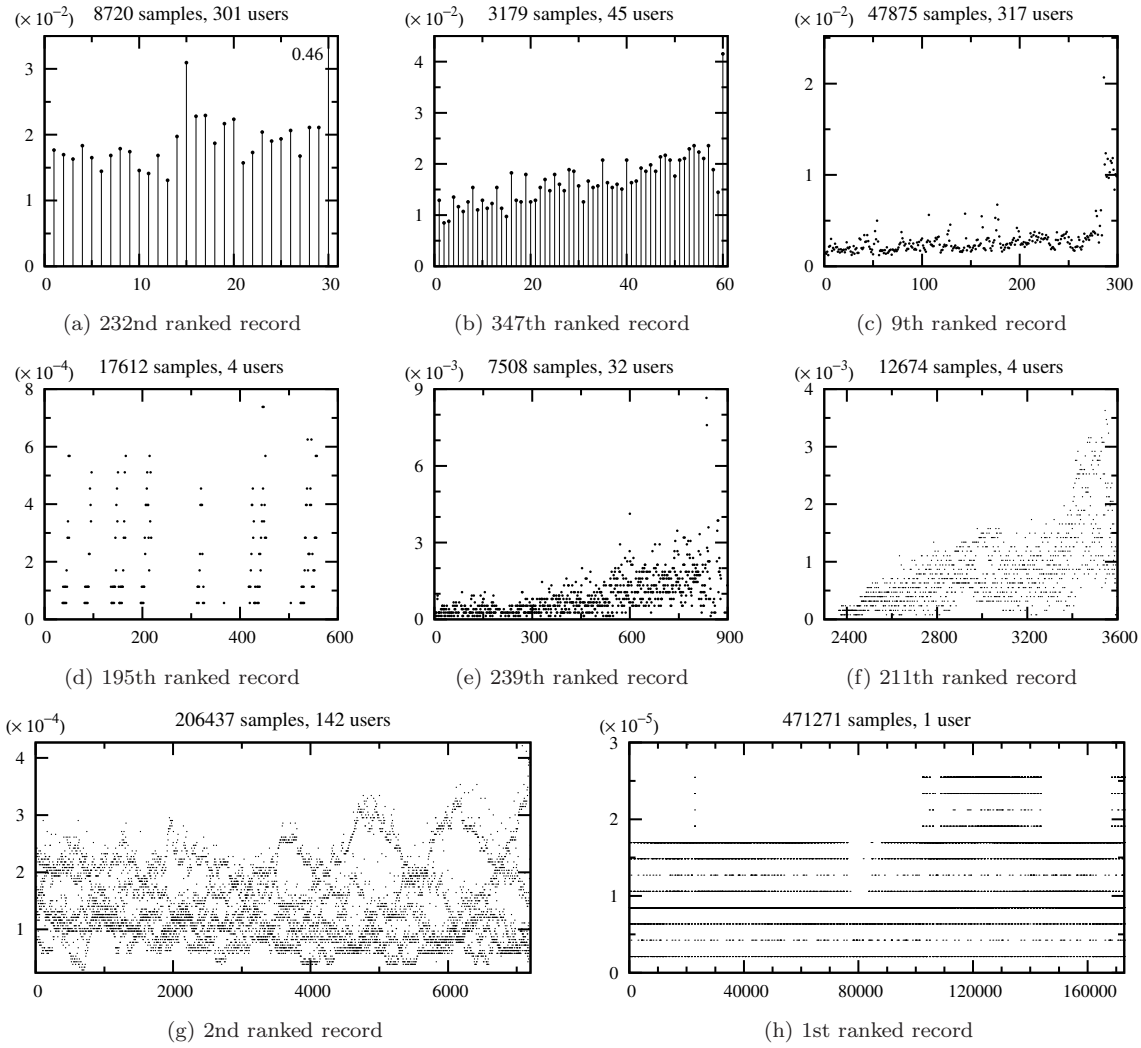


Figure 8: Probability mass function of TTL with initial TTL equal to (a) 30 s, (b) 60 s, (c) 300 s, (d) 600 s, (e) 900 s, (f) 3 600 s, (g) 7 200 s, and (h) 172 800 s (48 h).

A careful analysis of this trace reveals the following. First, requests instants and final responses instants do not differ much, thereby justifying our instantaneous transmission/processing assumption. Second, based on the TTLs recorded, Inria's DNS server respects the TTL rule. We are therefore in the case of a single traditional DNS cache. Third, the TTLs found in the final response packets (those from Inria's DNS server to users) vary from 1 to the initial TTL advocated by authoritative servers. For a given content, the initial TTL did not change for the duration of the trace, therefore the caching duration at the DNS server, for each content, has a deterministic distribution.

As the TTL in the cache decreases linearly with time, the requests from users get a value that decreases over time until the content is removed from the cache and a miss occurs. The TTL values found in the final response packets are samples from a reverse sawtooth wave. If users' requests form a Poisson process, then the TTL values will be uniformly spaced between 1 and the initial TTL.

The trace has initial TTL values in the set {30 s, 60 s, 300 s, 600 s, 900 s, 1 h, 2 h, 4 h, 12 h, 24 h, 48 h, 168 h}. We have checked the distribution for many contents and for initial TTL values in the subset {30 s, 60 s, 300 s, 600 s, 900 s, 1 h, 2 h, 48 h}. The missing values had not enough samples for any content with

that TTL value. We report the probability mass function in Fig. 8 for those contents that have the highest amount of samples of TTL for the respective initial value. It is clear that the distribution is not uniform in any of the depicted cases. Observe that in Figs. 8a-8g the number of users requesting the content is higher than one, consequently the samples that each user gets have a different distribution than the one depicted. In the case of Fig. 8h corresponding to the most popular content, the requests came from a single user, and therefore the depicted distribution is that seen by the cache of that single user and this distribution is not uniform. This analysis emphasizes the pertinence of our models and the need to consider for the TTLs as general distributions as possible.

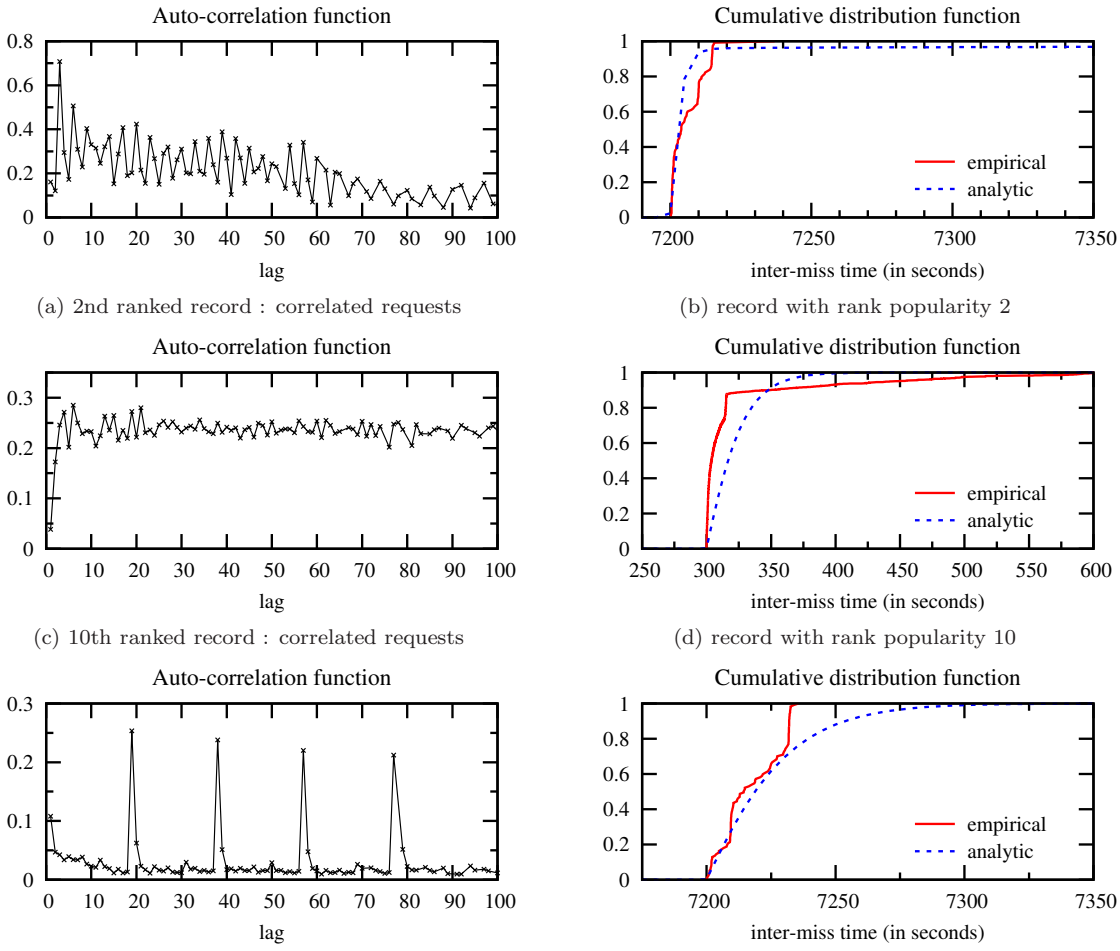
An important point that we considered was to check whether requests are correlated or not. We checked the auto-correlation function (ACF) of the requests of over 40 records having a large enough number of requests (this corresponds to a popularity rank below 350). We selected in particular those with TTL less than 7 200 to ease the subsequent computation of our models. Table 2 reports important characteristics of some of these records, namely the number of users (distinct requesters), the aggregate arrival rate, the TTL of each record in the cache and the maximum and the minimum of the ACF, together with the respective lags.

Table 2: Several characteristics of some of the resource records

Record rank	Number of users	Arrival rate (requests/s)	TTL (s)	Maximum of the ACF	Lag	Minimum of the ACF	Lag
2	142	0.23959	7 200	0.70766	3	0.01848	17 924
9	317	0.05556	300	0.39636	8	0.03796	1
10	215	0.03442	300	0.28527	6	0.03850	1
24	3	0.03250	7 200	0.13669	20	−0.02570	189
26	3	0.03243	7 200	0.03809	20	−0.01261	174
29	4	0.03233	7 200	0.06728	20	−0.01584	125
31	3	0.03232	7 200	0.04351	20	−0.01169	1 414
33	4	0.03232	7 200	0.40307	20	−0.02839	512
34	2	0.03231	7 200	0.19227	20	−0.01614	1 206
37	3	0.03230	7 200	0.59283	20	−0.04151	275
38	4	0.03230	7 200	0.04294	20	−0.01190	166
40	4	0.03229	7 200	0.42250	20	−0.03986	44
42	3	0.03228	7 200	0.53545	20	−0.03773	191
44	2	0.03227	7 200	0.02921	20	−0.00962	137
90	3	0.03058	7 200	0.25333	19	−0.00199	124
138	2	0.02930	7 200	0.06785	18	−0.03172	114
157	2	0.02924	7 200	0.02623	18	−0.01505	33
196	6	0.02043	600	0.22308	1	−0.00244	101
346	45	0.00368	60	0.17712	6	−0.00282	40
348	74	0.00368	600	0.13295	3	−0.00229	39

We have observed time-varying behavior (week day/week-end, day/night) in the requests arrival processes of many records. Such records exhibit correlations as can be seen in Table 2: the maximum of the ACF is far from 0 (e.g. records with popularity 2, 37 or 42). The ACF of requests for other records remains within 10% (e.g. records with popularity 26, 29, 31). The requests are positively correlated as the minimum of the ACF is close to 0. We conclude that Assumption 5 (renewal request process) is often not met. *Testing our model using records of this trace that have correlated requests will give insight into its robustness* since the main assumptions used in the single cache analysis will not be met.

Our aim is to predict the cache performance metrics and most importantly the cache miss process as it represents the traffic that flows upstream in the DNS hierarchy (also needed for network analysis). We have tested our model with the same records whose auto-correlation function was checked, we will however report the results on only three of them, which have different popularity ranks and different ACF shapes.



(a) 2nd ranked record : correlated requests

(b) record with rank popularity 2

(c) 10th ranked record : correlated requests

(d) record with rank popularity 10

(e) 90th ranked record : periodic correlation

(f) record with rank popularity 90

Figure 9: (a), (c), (e): correlation in requests; (b), (d), (f): miss process prediction.

Table 3: Mean, variance, squared coefficient of variation (SCV), and skewness of the inter-request times of selected records

Record rank	Mean (s)	Variance (s^2)	SCV	Skewness	#states of fitted MAP
2	4.16205	4477.5619	258.47	83.8751	128
10	29.06918	3836.6849	4.54	3.0162	16
90	32.70701	18.7103	18.71	20.7823	128

For each record, we used the KPC-Toolbox [31] to find the Markovian Arrival Process (MAP) that best fits the inter-request times X of the aggregated arrival process. This tool matches with priority higher-order correlations and can convert any MAP into a renewal process having inter-arrival times identically distributed as arrivals in the MAP. The number of states of the fitted MAP and the mean/variance/skewness of the empirical inter-request time (as computed by the tool) for records with ranks 2, 10, and 90 can be found in Table 3. These records appear to be significantly diverse as can be seen from their ACFs depicted in Figs. 9a, 9c, and 9e. Moreover, these records have a significant amount of requests and miss requests which allows to better compute the empirical distributions and to find a better renewal process fitting the arrival data using the KPC-Toolbox.

For each selected record (i.e. content), taking as input the fitted distribution and the TTL value, we use the findings of Section 4.2.1 (deterministic TTL) to obtain the performance metrics of the cache relative to

Table 4: Performance metrics and relative errors

Record rank	Miss rate			Hit probability			Occupancy		
	Trace	Model	Error (%)	Trace	Model	Error (%)	Trace	Model	Error (%)
2	0.00014	0.00014	0.920	0.99943	0.99941	0.002	0.99914	0.98995	0.920
10	0.00313	0.00311	0.765	0.90901	0.90924	0.025	0.93491	0.93516	0.027
90	0.00014	0.00014	0.114	0.99547	0.99544	0.003	0.99757	0.99944	0.188

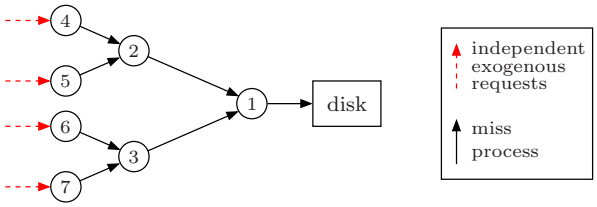


Figure 10: A binary tree with 7 caches.

this content (see Table 4) and the CDF of the inter-miss times (see Figs. 9b, 9d, and 9f). To determine the CDF (23), we use a naive Riemann's sum for the integral computation. Two parameters must be set: (*i*) the upper bound of the integral τ, and (*ii*) the step length Δ. Clearly, having a larger τ and a smaller Δ decreases the numerical error but increases the computational cost. We set $\Delta = 0.1$ and τ equal to 100 times the maximum between the mean inter-request time and the TTL. (For this reason, we tested our model on records having a TTL less than 7200 s.)

The analytic results for the performance metrics are compared to those computed from the trace. Table 4 reports *negligible values of the relative errors on the performance metrics (less than* 1%*)*. Looking now at the CDFs of both empirical and analytic results depicted in the right-hand side of Figure 9, we can say that *our model accurately estimates the CDF of the inter-miss time.*

We conclude from the above that *Corollary 1 and Proposition 3 appear to be applicable even if Assumption 5 is not met.* Recall Proposition 1; the expressions of the performance metrics are valid when requests form simply a *stationary and ergodic* point process. This section suggests that *our single cache model is robust.*

6.2. Network of Caches Case

We now proceed to evaluating the robustness of our model of a network of caches. It is worth recalling that with exponentially distributed caching durations our model coincides with the one developed in [11] to study caches that reset the caching durations at each hit. In [11], Approximation 1 is also used; the authors evaluate their model by comparing the approximate results it yields to exact analytic results that can be found when the conceptual IRM is used for requests. An excellent match is found which legitimates the use of Approximation 1. In other words, *our model approximates very well a network of caches with IRM requests and exponentially distributed caching durations.*

To test out model under other conditions, we resort to performing simulations. We will consider at first a situation in which Assumptions 4-5 are satisfied, which can be achieved using event-driven simulations. This allows to test the quality of Approximation 1. Simulated results are used as "ground truth" to evaluate the goodness of our approximate results (see Section 6.2.1). Seeing that our model is extremely accurate, we test our model in a situation where Assumption 5 is not met, more precisely, inter-request times will be correlated. This can be achieved by performing trace-driven simulations using the traces collected at Inria. We report the results in Section 6.2.2.

In both cases, we consider a binary tree consisting of 7 caches as shown in Fig. 10. This tree represents well the hierarchy found in DNS: cache 1 is that of the authoritative server, caches 2 and 3 are typically those of ISP's DNS servers, and caches 4-7 are found at the client side (ADSL modem, laptop, etc.). However, unlike the DNS hierarchy where nodes have a high degree, the node degree is 2 in our simulated network. While we chose a binary tree to ease the computation, it is interesting to note that it is the most challenging

case for Approximation 1, as it is expected to yield more accurate results as the node degree increases (see the discussion after the statement of Approximation 1).

6.2.1. Testing Quality of Approximation 1 Using Event-Driven Simulations

We have performed event-driven simulations in which requests were generated according to a predetermined distribution and different distributions for the caching durations were simulated.

To capture the fact that users have interleaving activity and inactivity periods, we assumed that requests for all contents form a Markov-Modulated Poisson Process (MMPP). In other words, we considered an Interrupted Poisson Process (IPP) to simulate the requests for a single content at each bottom-level cache. The request rate at cache i is $\lambda_i \in [0.5, 20]$ for $i \in \{4, 5, 6, 7\}$. As a consequence of IPP requests, each component (miss process) of the overall request process at caches 2, 3 and subsequently 1 is not a Poisson process. This setting enables to test the quality of Approximation 1.

The caching durations at all caches follow the same distribution, with expectation in $[0.5, 1.5]$. Four distributions have been considered in the simulations: deterministic, hypo-exponential, exponential and hyper-exponential. Their respective coefficients of variation are 0, < 1, 1, and > 1. This allows to observe the effect of increasing the randomness of the TTL.

The "exact" values of the performance metrics are those obtained after running long enough simulations. Our criterion for a long simulation is one that yields a relative incertitude on each metric less than 10^{-4}. For instance, the hit probability at cache i obtained through simulation is $h^S_{P,i}$ (the superscript S stands for "simulation"). We calculated the 99% confidence interval $[h^S_{P,i} - \epsilon, h^S_{P,i} + \epsilon]$, the relative incertitude on $h_{P,i}$ is then $2\epsilon/h^S_{P,i}$. At the end of a simulation run, the latter was at most 0.6×10^{-4}.

The approximate values of the performance metrics are those predicted by our model and are obtained by following the recursive procedure explained in Section 5.3. We have implemented a MATLAB numerical solver that determines the CDFs in the network (using (31)-(32)) and then the metrics of interest at each cache (using Corollary 1 where $\mathbb{E}[Z_c] = \lim_{t\to\infty} L_c(t)$). The numerical error comes from the integral computation used in (31)-(32) (e.g., the integrals over infinite ranges). Again, we use Riemann's sum and, for simplicity, unique values for τ and Δ for all computations relative to a single simulation run. Consider all inter-request times and all caching durations within the network of caches. We set τ to one hundred-fold the *maximum* expectation among all these rvs, and Δ to one thousandth of the *minimum* expectation among the same rvs.

Remark 4. *The IPP is stochastically equivalent to a renewal process with a two phase hyper-exponential CDF (also known as the H_2 distribution) which is a special case of the diag.ME distribution. Also, three out of the four distributions considered for the caching durations, namely the hypo-exponential, the exponential and the hyper-exponential distributions, are also special cases of the diag.ME distribution. Therefore, three out of the four simulated scenarios fall in the context of Proposition 10. Instead of following the general recursive procedure, we could have determined the CDFs in the network using the closed-form equations* (43) *and* (51) *and then the metrics of interest at each cache using Corollary 1 and* (26)*. The numerical error in this case is due to finding numerically the roots of* (39) *and the solution of* (40)*.*

We have computed the relative error between the exact results obtained from simulations and the approximate results predicted by our model. The average relative error across all simulations on the miss rate, the hit probability and the occupancy at caches from different hierarchical levels are reported in Table 5 (columns 4, 6, 8, and 10). Our model is extremely accurate in predicting the performance metrics when caching durations are not deterministic as the relative error does not exceed 0.3%. For deterministic caching durations, an excellent prediction is available at bottom-level caches. The relative error increases as we consider caches at higher hierarchical levels, it reaches roughly 5% at the third level, which is nevertheless an affordable value.

We performed another series of simulations which is essentially the same as the series described above with the exception of having deterministic TTLs at all bottom-level caches for all four TTL distributions considered at higher-level caches. At first we used the same values of λ_i for $i \in \{4, 5, 6, 7\}$ as in Table 5, then we repeated the simulations using another set of values that is $\lambda_4 = 0.052$ requests/s, $\lambda_5 = 0.061$ requests/s,

Table 5: Analytic performance metrics and their relative errors (in percentage) at representative caches ($\lambda_4 = 1.57$ requests/s, $\lambda_5 = 0.87$ requests/s, $\lambda_6 = 1.37$ requests/s, $\lambda_7 = 0.68$ requests/s)

Cache	Performance metric	Distribution of caching durations								Trend
		deterministic		hypo-exponential		exponential		hyper-exponential		
		value	rel. err.	value	rel. err.	value	rel. err.	value	rel. err.	
4	miss rate	0.4948	0.0092	0.4991	0.0065	0.5004	0.0872	0.5024	0.0770	↗
	hit probability	**0.4328**	0.0383	0.4279	0.0272	0.4263	0.0066	0.4241	0.0007	↘
	occupancy	0.3579	0.0447	0.3609	0.0471	0.3619	0.0336	0.3633	0.0236	↗
2	miss rate	0.5671	1.1214	0.5267	0.0848	0.5168	0.1026	0.5107	0.0013	↘
	hit probability	0.4161	1.4561	0.4639	0.1868	0.4759	0.1514	**0.4841**	0.1032	↗
	occupancy	0.5817	1.1460	0.5402	0.0485	0.5301	0.0631	0.5238	0.0418	↘
1	miss rate	0.5293	5.0614	0.4823	0.2367	0.4697	0.0687	0.4605	0.0065	↘
	hit probability	0.5179	4.5360	0.5205	0.2525	0.5236	0.1067	**0.5273**	0.0707	↗
	occupancy	0.6767	5.0986	0.6167	0.1965	0.6005	0.0277	0.5887	0.0366	↘

$\lambda_6 = 0.091$ requests/s, $\lambda_7 = 0.078$ requests/s. For both sets of values, the results are very similar to those reported in Table 5 and are not reported here to avoid repetitions.

Our comparative study suggests that *using Approximation 1 is not a limitation.*

Remark 5. *An interesting trend is noted in the values reported in Table 5. (The same trend is seen in the results of the other series of simulations not reported here.) Considering a given metric (m_R, h_p or π), its value changes monotonically with the coefficient of variation of the caching durations' distribution. The trend of each metric and at each cache is shown in Table 5, column 11. It is important to observe that* no distribution achieves the maximum hit probability at any cache *(values in bold font in Table 5).*

This remark suggests that for IPP requests, *deterministic caching durations should be used only at bottom-level caches*, i.e., at the client side. *Caches at servers should store contents for durations as variable as possible (large coefficient of variation)* in order to improve the hit probability. Investigating whether this remark holds under different traffic conditions is left for future work.

6.2.2. Testing Robustness to Assumption 5 using Trace-Driven Simulations

In this section, we assess the robustness of our model (and therefore our renewal assumptions) under realistic traffic conditions. We repeat the previous experiment on the binary tree (see Fig. 10), but instead of generating requests synthetically, we rely on users' requests from the trace collected at the Inria DNS cache (see Section 6.1 for a description of this trace).

In this trace all requests for a content form a single stream that arrives to a single cache. What we need instead are four streams of requests for the same content, one stream for each bottom-level cache. We resort to artificially separate one single stream of requests into four streams. Naturally, one would pick the most popular content which has the highest amount of requests to do this manipulation. However, the content ranked first had requests issued by a single user. Splitting these requests into four streams would result in four correlated request streams, whereas we need the input requests at bottom-level caches to be independent.

Fortunately, the data related to the 2nd most popular content aggregates 206 437 requests issued by 142 distinct users. This data is thus suitable for our experiment. We split the data into four sets and use each set to reproduce the request arrival process at one particular bottom-level cache. Doing so, the following holds:

- 67 079 requests issued by one user arrive to cache 4;
- 56 683 requests issued by one user arrive to cache 5;
- 41 338 requests issued by seventy users arrive to cache 6;
- 41 337 requests issued by seventy users arrive to cache 7.

Table 6: Mean, variance and SCV of the inter-request times at caches 4, 5, 6 and 7

Bottom-level cache	Number of users	Number of requests	Inter-request times		
			Mean (s)	Variance (s^2)	SCV
4	1	67079	8.10881	1821380.58	27700.42
5	1	56683	3.10663	195484.85	20255.08
6	70	41338	20.84186	77033.81	177.34
7	70	41337	20.84120	33448.23	77.01

Table 7: Mean TTL at all caches

Cache	1	2	3	4	5	6	7
Mean TTL	1.27900	1.02600	1.00540	1.28340	0.72360	1.44790	1.02780

The mean/variance/SCV of the inter-request time at each bottom-level cache can be found in Table 6. Each sub-trace/input process is fitted into a MAP using KPC-Toolbox and the CDF of the first inter-request time is used as the input of our TTL-based cache network model.

Regarding the TTL distribution, we consider that timers are either all deterministic or all exponentially distributed with mean uniformly chosen at random in $[0.5, 1.5]$ seconds. The values used are in Table 7.

Our objective is to compare the analytic results predicted by our network model to the results found by simulations. We have modified the event-driven simulator used in Section 6.2.1 to use the four sets of requests at bottom-level caches instead of generating IPP synthetic requests. This trace-driven simulator was run for a long enough time in order to have a relative incertitude on each metric less than 10^{-4}.

As for the analytic results, we implemented again a MATLAB numerical solver that applies the recursive procedure presented in Section 5.3. The solver computes sequentially:

1. the CDFs of the miss processes at caches 4 to 7 (results similar to those in Figs. 9b, 9d and 9f);
2. the CDFs of the aggregate requests at caches 2 and 3 (results in Figs. 11a and 11c);
3. the CDFs of the miss processes at caches 2 and 3 (results in Figs. 11b and 11d);
4. the CDF of the aggregate requests at cache 1 (result in Figure 11e);
5. the CDF of the miss process at cache 1 (result in Figure 11f).

Numerical errors or model mispredictions propagate upwards in the tree of caches, as the arrival process at higher caches builds upon the miss process from previous caches.

The results of the comparison of the analytic and empirical distributions of the miss processes at caches 4 to 7 are similar to those in Figs. 9b, 9d and 9f. Regarding caches 3 to 1, we report in Figure 11 the analytic and empirical distributions when timers are exponentially distributed.

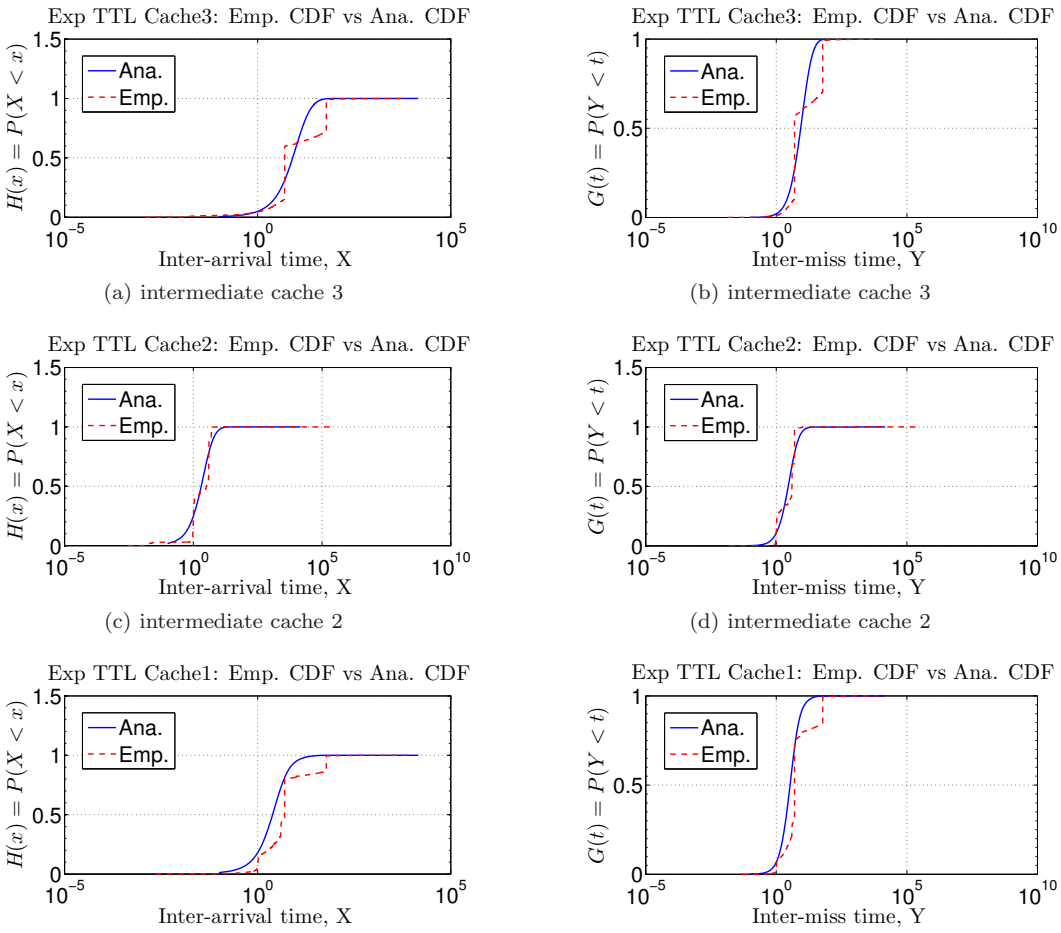
The CDFs of the interval of the aggregate requests processes are depicted in Figs. 11a, 11c and 11e for caches 3, 2, and 1 respectively. Our model computes the CDF according to (31) assuming that requests processes at caches 4 to 7 are renewal processes. Despite this assumption being violated in the traces used, distributions in every graph are fairly close to each other.

Recall that (31) is the CDF of the *first* inter-request time of the aggregate process. According to Approximation 1, we use this CDF for all subsequent inter-request times (renewal approximation).

The CDFs of the inter-miss times, both empirical and analytic, at caches 3, 2 and 1 are displayed in Figs. 11b, 11d and 11f respectively. The analytic CDF is derived according to (32) using Approximation 1. The CDF used for the aggregate process is the analytic one and not the empirical one. Overall, we find that the results shown in Fig. 11 are promising, encouraging us to pursue the validation of our model to cover other distributions for the caching durations.

7. Conclusions

The analytic models introduced in this paper allow to study the modern DNS cache hierarchy. Our single cache model has been tested on real DNS traces that do not meet the renewal assumption. It predicts



(a) intermediate cache 3 (b) intermediate cache 3

(c) intermediate cache 2 (d) intermediate cache 2

(e) root cache 1 (f) root cache 1

Figure 11: Empirical and analytic distributions for the aggregate request processes and the miss processes at caches 3, 2 and 1.

the performance metrics and the CDF of the miss process remarkably well. The approximation used in our network of caches model has been validated through simulations. We have found that if the renewal function of the request process is concave, then the deterministic policy maximizes the hit probability and minimizes the occupancy. For convex renewal functions, the opposite holds. Our numerical analysis suggests that no distribution achieves the maximum hit probability at all caches in a hierarchical network.

Acknowledgments

The authors would like to thank Francis Montagnac and Marc Vesin (IT staff at Inria, Sophia Antipolis) for collecting and anonymizing the DNS traces. The authors are deeply grateful to Fabrice Huet for his help in processing the large amount of data collected. The authors thank the referees for their valuable comments.

References

[1] N. Choungmo Fofack and Sara Alouf, Modeling modern DNS caches, in: Proc. ACM ValueTools '13, Torino, Italy, 2013.

[2] J. Pang, A. Akella, A. Shaikh, B. Krishnamurthy, S. Seshan, On the responsiveness of DNS-based network control, in: Proc. IMC '04, Taormina, Italy, 2004.

[3] R. J. Bayardo, R. Agrawal, D. Gruhl, A. Somani, YouServ: a web-hosting and content sharing tool for the masses, in: Proc. ACM WWW '02, New York, USA, 2002, pp. 345–354.
[4] T. Callahan, M. Allman, M. Rabinovich, On modern DNS behavior and properties, ACM SIGCOMM Computer Communication Review 43 (3) (2013) 7–15.
[5] Y. T. Hou, J. Pan, K. Sohraby, S. X. Shen, Coping miss synchronization in hierarchical caching systems with nonlinear TTL functions, in: Proc. IEEE ICC '04, 2004, pp. 2194–2198.
[6] N. Choungmo Fofack and Sara Alouf, Non-renewal TTL-based cache replacement policy and applications: Case of modern DNS hierarchy, Tech. Rep. RR-8414, Inria (Nov. 2013).
[7] Y. T. Hou, J. Pan, B. Li, S. Panwar, On expiration-based hierarchical caching systems, IEEE Journal on Selected Areas in Communications 22 (1).
[8] J. Jung, A. W. Berger, H. Balakrishnan, Modeling TTL-based Internet caches, in: Proc. IEEE Infocom '03, San Francisco, CA, USA, 2003.
[9] O. Bahat, A. M. Makowski, Measuring consistency in TTL-based caches, Performance Evaluation 17 (2005) 439–455.
[10] V. Martina, M. Garetto, E. Leonardi, A unified approach to the performance analysis of caching systems, in: Proc. IEEE Infocom '14, Toronto, Canada, 2014.
[11] N. Choungmo Fofack, P. Nain, G. Neglia, D. Towsley, Analysis of TTL-based cache networks, in: Proc. ACM ValueTools '12, Cargèse, France, 2012.
[12] N. Choungmo Fofack, P. Nain, G. Neglia, D. Towsley, Performance evaluation of hierarchical TTL-based cache networks, Computer Networks 65 (2014) 212–231.
[13] N. Choungmo Fofack, On models for performance analysis of a core cache network and power save of a wireless access network, Ph.D. thesis (Feb. 2014).
[14] D. S. Berger, P. Gland, S. Singla, F. Ciucu, Exact analysis of TTL cache networks, Performance Evaluation 79 (2014) 2–23.
[15] J. Jung, E. Sit, H. Balakrishnan, R. Morris, DNS performance and the effectiveness of caching, in: Proc. ACM SIGCOMM Workshop on Internet Measurement (IMW '01), New York, NY, USA, 2001.
[16] F. Baccelli, P. Brémaud, Elements of Queueing Theory, Palm Martingale calculus and Stochastic recurrences, 2nd Edition, Springer, Berlin, 2003.
[17] W. Whitt, Approximating a point process by a renewal process, i: Two basic methods, Operations Research 30 (1) (1982) 125–145.
[18] A. Feldmann, W. Whitt, Fitting mixtures of exponentials to long-tail distributions to analyze network performance models, in: Proc. IEEE Infocom '97, Kobe, Japan, 1997.
[19] D. R. Cox, Théorie du Renouvellement, Monographies Dunod, Paris, 1966.
[20] M. Tortorella, Numerical solutions of renewal-type integral equations, INFORMS Journal on Computing 17 (2005) 73–96.
[21] S. Karlin, Total Positivity, Vol. 1, Stanford University Press, 1968.
[22] M. Brown, Bounds, inequalities, and monotonicity properties for some specialized renewal processes, The Annals of Probability 8 (2) (1980) 227–240.
[23] A. Dan, D. Towsley, An approximate analysis of the LRU and FIFO buffer replacement schemes, in: Proc. ACM Sigmetrics '90, Boulder, CO, USA, 1990, pp. 143–152.
[24] E. J. Rosensweig, J. Kurose, D. Towsley, Approximate models for general cache networks, in: Proc. IEEE Infocom '10, San Diego, USA, 2010.
[25] A. Simonian, M. Gallo, B. Kauffmann, L. Muscariello, C. Tanguy, Performance of the random replacement policy for networks of caches, in: Proc. ACM Sigmetrics/Performance '12, London, UK, 2012, pp. 395–396.
[26] J. A. Fill, L. Holst, On the distribution of search cost for the move-to-front rule, Random Structures Algorithms 8 (3) (1996) 179–186.
[27] S. Karlin, H. M. Taylor, A First Course in Stochastic Processes, 2nd Edition, Elsevier, 1975.
[28] A. T. Lawrance, Dependency of intervals between events in superposition processes, Journal of the Royal Statistical Society, Series B (Methodological) 35 (2) (1973) 306–315.
[29] V. Isham, Dependent thinning of point processes, Journal of Applied Probability 17 (4) (1980) 987–995.
[30] A. D. Polyanin, A. V. Manzhirov, Handbook of Integral Equations, 1st Edition, CRC Press, 1998.
[31] G. Casale, E. Zhang, E. Smirni, KPC-Toolbox: Simple yet effective trace fitting using Markovian Arrival Processes, in: Proc. QEST '08, 2008.