Non-renewal TTL-based Cache Replacement Policy and Applications: Case of Modern DNS Hierarchy

Nicaise Choungmo Fofack, Sara Alouf

▶ **To cite this version:**

Nicaise Choungmo Fofack, Sara Alouf. Non-renewal TTL-based Cache Replacement Policy and Applications: Case of Modern DNS Hierarchy. [Research Report] RR-8414, INRIA. 2013. hal-00914877

HAL Id: hal-00914877

https://inria.hal.science/hal-00914877

Submitted on 6 Dec 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

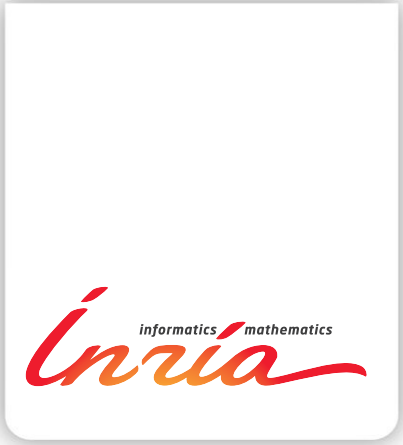
L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Non-renewal TTL-based Cache Replacement Policy and Applications: Case of Modern DNS Hierarchy

Nicaise Choungmo Fofack, Sara Alouf

RESEARCH REPORT
N° 8414
December 2013
Project-Team Maestro

ISSN 0249-6399 ISRN INRIA/RR--8414--FR+ENG



Non-renewal TTL-based Cache Replacement Policy and Applications: Case of Modern DNS Hierarchy

Nicaise Choungmo Fofack*†, Sara Alouf †

Project-Team Maestro

Research Report n° 8414 — December 2013 — 24 pages

Abstract: Caching is undoubtedly one of the most popular solution that easily scales up with a world-wide deployment of resources. Records in Domain Name System (DNS) caches are kept for a pre-set duration (time-to-live or TTL) to avoid becoming outdated. *Modern* caches are those that set locally the TTL regardless of what authoritative servers say. In this report, we introduce analytic models to study the modern DNS cache behavior based on renewal arguments. For both the single cache case and the network of caches case, we derive the cache performance metrics and characterize at each cache the miss process and the aggregate request process. We address the problem of the optimal caching duration and find that if inter-request times have a concave CDF, then the deterministic policy is the best. We validate our single cache model using real DNS traces and our network of caches model using event-driven simulations. Our models consider general caching durations and are tested with deterministic, hypo-exponential, exponential and hyper-exponential distributions. Our models are very robust as the relative error between empirical and analytic values stays within 1% in the first case and within 5% at the highest cache level in the network case. Our models successfully predict the CDF of the miss process even when the renewal assumption is not met.

Key-words: Cache replacement policy, cache architecture, Renewal theory, timer, time-to-live (TTL), Domain Name Service (DNS), content-centric network.

* This author was supported by the University Nice Sophia Antipolis under a grant from the French Government.

† Email: {nicaise.choungmo_fofack, sara.alouf}@inria.fr. Postal address: Inria, B.P. 93, 06902 Sophia Antipolis, Cedex, France.

RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Modèle de gestion de caches à base de durées de vie non-regénérées : Cas des réseaux de caches DNS dits modernes

Résumé : Le cache est sans doute l'une des solutions les plus populaires et les mieux adaptées pour un déploiement mondial de ressources. Les enregistrements concernant les noms de domaines sur Internet sont conservés dans les caches DNS (Système de noms de domaine) pour une durée prédéterminée nommée TTL (*time-to-live*), évitant ainsi l'obsolescence des enregistrements dans le cache. Les caches DNS dits *modernes* implantent leur propre valeur TTL, indépendamment de la valeur recommandée par les serveurs de noms autoritaires du domaine concerné. Dans ce rapport, nous nous appuyons sur la théorie du renouvellement pour développer des modèles analytiques afin d'étudier les DNS modernes. Nous calculons les performances d'un cache en termes d'occupation et des probabilités de *hit*/*miss* et caractérisons le processus en sortie du cache (le processus des *miss*). Ces résultats, obtenus d'abord pour un cache isolé, sont par la suite étendus aux cas d'un réseau de caches. Dans ce dernier cas, nous caractérisons également le processus issu de l'agrégation des requêtes dans les caches de niveau supérieur. Nous abordons le problème de la distribution optimale du TTL dans un cache et trouvons que si les temps inter-requêtes ont une fonction de répartition concave alors l'optimalité est obtenue en imposant un TTL déterministe. Nos résultats pour un cache isolé sont validés sur une trace réelle de trafic DNS et ceux pour un réseau de caches sont validés par des simulations à événements discrets, considérant des TTLs de distribution déterministe, hypo-exponentielle, exponentielle ou hyper-exponentielle. Nos modèles s'avèrent être très robustes puisque l'erreur relative entre les valeurs empiriques et analytiques reste inférieure à 1% dans le cas d'un cache isolé et à 5% dans le cas du réseau, chez le cache de plus haut niveau. Ainsi, même si le processus de requêtes n'est pas de renouvellement, notre modèle donne avec précision la distribution du processus en sortie d'un cache.

Mots-clés : Politiques de caches, réseau de caches, théorie du renouvellement, temporisateur, durée-de-vie (TTL), service de noms de domaine (DNS), réseau orienté contenus.

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 [15], the World Wide Web [6], Content Distribution Networks [21], or the recently proposed Information-Centric Network (ICN) architectures [1]. 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 replacement (RND), etc.

In this report, 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.

The Domain Name System (DNS) is a valid application case. When a site's name is typed into a browser, DNS looks up the corresponding IP address for that site. To do so, 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.

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, called the *TTL rule* in the literature, 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. Records marked with a null value should not be cached. Caches compliant with the TTL rule are referred to as *traditional DNS caches*. Those overriding the advocated TTL with a locally chosen value (cf. [18, 3]) are called *modern DNS caches* [4].

The objective of this report is to assess the performance of modern DNS caches. The contributions of this work are as follows:

- we are the first to provide analytic models to study both a single modern DNS cache and a network of (modern) caches with general caching durations;
- we characterize the distribution of the DNS traffic flowing upstream in the DNS hierarchy besides deriving the usual cache performance metrics;
- for the case of a single cache we identify when is the deterministic caching duration the optimal policy and discuss the optimal deterministic value when this is the case;
- for the case of a network of caches with diagonal matrix-exponential distributions, we compute the distribution of the request and miss processes anywhere in the network in closed-form;

- we check the robustness of our single cache model over DNS traces collected at Inria and
- the robustness of our network of caches model through event-driven simulations.

The rest of the report is organized as follows. Section 2 reviews the works most relevant to this report. Section 3 presents the scenario considered and most of the definitions, assumptions and notation used throughout the report. Our single cache model is analyzed in Sect. 4 and its applicability in other contexts is discussed in Sect. 5. The case of a network of caches is analyzed in Sect. 6. We validate our models in Sect. 7 and show some numerical results. Section 8 summarizes our findings.

2 Related Works

2.1 TTL rule and caching durations

By marking R with a suitable caching duration, the authoritative server makes sure that no outdated copy is cached. A consequence of the TTL rule is that all copies of a record along a path would be cached mainly for the same duration. 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. Cache misses are then *synchronized* [13], this reflects a reduced efficiency of caching.

Short caching durations translate in a good consistency between the original version of a record and its copies [12, 15] and balance the load between servers, avoiding congestion at bottom level content provider's servers [18]. The downside is an increase of the DNS traffic even if records remain unchanged at the authoritative servers. Short caching durations emphasize also the so-called *miss synchronization effect* [13].

Recently, some experiments on Internet led to the observation that the TTL rule is not always applied [4, 18]. Records may be cached for a locally chosen duration regardless of the TTL value marked on them by the answerer. Some DNS server software can be configured so that DNS caches override low TTL values with a global minimum duration for instance. Bayardo *et al.* mention in [3] that many servers at IBM are apparently configured to cache records for at least five minutes whereas browsers like Mozilla and most likely Internet Explorer use a default caching duration (fifteen minutes for Mozilla). Jackson *et al.* explained later in [14] that, by doing so, web browsers protect users from cross-site scripting attacks. Jung, Berger and Balakrishnan have shown experimentally in [15] that fifteen minutes long caching achieves a global cache hit ratio over 80%. Breaking the TTL rule and caching for longer durations do have advantages at the risk of making a site inaccessible to the client for some minutes.

2.2 State-of-the-art in modeling DNS caches

Since the recent observation of the modern behavior of DNS caches [4, 18], only few results of the state of the art are applicable to modern DNS caches. Hou *et al.* consider in [12] a tree of traditional DNS caches fed by Poisson traffic. The performance metrics derived in [12] 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 [15] 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 [15] 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.

The closest paper to our work, methodologically speaking, is [7]. Choungmo *et al.* analyze both a single cache and a network of caches in which each content remains in cache for a random period. 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 [7] 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 [7] is that of ICN achitectures.
It has been reported in [4, 16, 18]—and we have observed it in our collected DNS traces—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 modern DNS cache. 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 will address a more general problem in this report, namely, finding the best distribution.

3 Scenario, Definitions, Assumptions

3.1 Considered Scenario

In this report, 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 [15]. A nice consequence is that the management of different records can safely be decoupled, simplifying thereby the modeling of caches. Our analysis will focus on a *single* content/record, characterizing the processes relevant to it, keeping in mind that the *same* can be *repeated* for every single content requested by users. This will be done in Sect. 5.2, where multiple files share a single limited buffer cache.

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 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. It initiates a new duration during which the content will be cached.

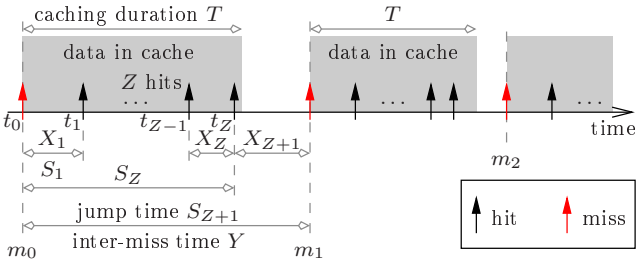


Figure 1: Requests, caching durations and inter-miss times.

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. Observe that the content in cache is updated only upon a cache miss. But it is only when the update originates from the authoritative server that one can absolutely be certain that the given update is correct. Therefore, a relevant performance metric is the *correctness* probability of a cache. Another property of a cache is its *freshness*. It defines how fast a change in a record can propagate until this cache. High freshness is desirable with dynamic authoritative 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 k-th 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) $f_{(k)}(t) = dF_{(k)}(t)/dt$. The arrival process is $\{N(t), t > 0\}$ with $N(t) = \sup\{k : S_k \leq t\} = \sum_{k>0} \mathbf{1}\{S_k \leq t\}$.

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.

When considering more than one cache, a subscript referring to the cache number will be added to the random variables for disambiguation.

Besides the "instantaneous transmission/processing" assumption that holds throughout this report, the following holds:

Assumption 1 (renewal arrivals) *Inter-request times are independent and identically distributed random variables.*

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 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 in agreement with the analysis in [15] and [10]. Feldmann and Whitt show in [10] that heavy-tailed processes can be well approximated by a renewal process with a hyper-exponential inter-arrival distribution. Jung, Berger and Balakrishnan show in [15] that the request process arriving at a DNS server's cache is heavy-tailed. Renewal processes with either Weibull or Pareto inter-event distributions are used to fit the collected inter-request times. Assumptions 2 and 4 hold at modern DNS servers [18, 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 *tree* network (that is a directed graph without any undirected cycles).

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. 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 report.

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 Sect. 6 when studying multiple caches in a tree network.

Table 1: Glossary of Main Notation

h_P	hit probability	$F(t)$	CDF of X
h_R	hit rate	$f(t)$	PDF of X
m_P	miss probability	λ	arrival rate ($1/\mathbb{E}[X]$)
m_R	miss rate	$N(t)$	requests during t (rv)
π	occupancy	$M(t)$	renewal function
T	caching duration (rv)	$m(t)$	renewal density funct.
$T(t)$	CDF of T	Y	inter-miss time (rv)
$1/\mu$	expectation of T	$G(t)$	CDF of Y
X	inter-request time (rv)	Z	hits during T (rv)
S_k	kth jump time (rv)	$\chi^*(s)$	LST of $\chi(t)$
$L(t)$	expected number of hits until t within T		
$H(t)$	CDF of inter-request time at higher-level cache		

4.1 The Model and its Analysis

Our first goal is to characterize the miss process which is the same as the process going out from a server towards the higher-level server. The request process and the caching durations are as assumed in Sect. 3, i.e. $\{N(t), t > 0\}$ is a renewal process. The renewal function and the renewal density function associated to $\{N(t), t > 0\}$ are, respectively, $M(t) = \mathbb{E}[N(t)] = \sum_{k>0} F_{(k)}(t)$ and $m(t) = \frac{dM(t)}{dt} = \sum_{k>0} f_{(k)}(t)$. It is well-known that the renewal function satisfies the so-called renewal equation [8]

$$M(t) = F(t) + \int_0^t M(t-x)dF(x) = F(t) + \int_0^t F(t-x)dM(x). \tag{1}$$

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, then following the definition of Sect. 3 we have that $Z = N(T)$ and its expectation is $\mathbb{E}[Z] = \mathbb{E}[N(T)] = \mathbb{E}\left[\mathbb{E}[N(T)|T]\right] = \mathbb{E}[M(T)]$ (M is a function).

Proposition 4.1 (Miss process) *Under Assumptions 1 and 2 the miss process of a single cache is a renewal process.*

Proof 4.1 *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 4.1 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. Deriving $G(t)$ completes 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 $t \geq 0$

$$L(t) = \sum_{k>0} \mathbb{P}(S_k < t, T > S_k) = \int_0^t \bar{T}(x)dM(x). \tag{2}$$

Observe that $L(\infty)$ is nothing but the expected number of hits in a renewal interval and is equal to $\mathbb{E}[Z]$.

Proposition 4.2 (Inter-miss times) *The CDF $G(t)$ of the generic inter-miss time Y and its LST are given by*

$$G(t) = F(t) - \int_0^t (1 - F(t-x))dL(x) \tag{3}$$

$$G^*(s) = 1 - (1 - F^*(s))(1 + L^*(s)). \tag{4}$$

Proof 4.2 *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

$$\begin{aligned} G(t) &= \sum_{k\geq 0} \int_0^t \int_0^{t-u} (T(u+x) - T(u)) f(x)dx \; f_{(k)}(u)du \\ &= \sum_{k\geq 0} \int_0^t \int_0^v (T(v) - T(u)) f(v-u) f_{(k)}(u)\, du\, dv \end{aligned}$$

The last equality is obtained after letting $v = u + x$ in the inner integral and then exchanging the integrals. Observe now that, under Assumption 1, the density $f_{(k)}(t)$ of the jump time S_k is the k-fold convolution of $f(t)$ (the density of X). Also, the convolution of $f_{(k)}$ and f is nothing but $f_{(k+1)}$. Note that $S_0 = 0$ and $f_{(0)}(t) = \mathbf{1}\{t = 0\}$. A straightforward calculation yields

$$\begin{aligned} G(t) &= \sum_{k>0} \int_0^t (1 - F(t-x)) T(x) f_{(k)}(x)dx \\ &= \int_0^t (1 - F(t-x))(1 - \bar{T}(x))dM(x) \\ &= F(t) - \int_0^t (1 - F(t-x))\bar{T}(x)dM(x) \end{aligned} \tag{5}$$

where we have used (1) to write (5). By differentiating (2) and using $dL(x)$ in (5), we find (3). It suffices to differentiate (3) then apply the Laplace transform to get the LST given in (4). The proof is complete.

Proposition 4.2 states that one needs to know the CDFs of the arrival process and the caching duration to derive the CDF of the miss process, or equivalently, the outgoing process. This proposition will be repeatedly used in Sect. 6 when analyzing networks of caches.

4.2 Performance Metrics

Our next goal is to derive the performance metrics defined in Sect. 3 at a single cache. Note that these metrics have been defined with respect to a single content. Similar metrics for the a set of contents can also be defined as long as the contents popularity is known. The following proposition provides the cache performance metrics.

Proposition 4.3 (Cache performance) *Under Assumption 1, the stationary hit probability h_P, the stationary miss probability m_P, the occupancy π, the stationary hit rate h_R, and the stationary miss rate m_R are respectively given by*

$$h_P = \frac{\mathbb{E}[Z]}{1+\mathbb{E}[Z]}\ ; \qquad m_P = \frac{1}{1+\mathbb{E}[Z]}\ ; \qquad \pi = \frac{\lambda/\mu}{1+\mathbb{E}[Z]}\ ;$$
$$h_R = \frac{\lambda\mathbb{E}[Z]}{1+\mathbb{E}[Z]}\ ; \qquad m_R = \frac{\lambda}{1+\mathbb{E}[Z]}.$$

Proof 4.3 *In the stationary regime, $\mathbb{E}[Z]$ is the expected number of hits within a renewal interval and $\mathbb{E}[Z]+1$ is the expected number of requests (including the single miss) in a renewal interval. Their ratio naturally gives the hit probability. We can readily find $m_P = 1 - h_P$, $h_R = \lambda h_P$ and $m_R = \lambda m_P$ since λ is the requests arrival rate. As Y is the inter-miss time, we have $\mathbb{E}[Y] = 1/m_R$. Last, regarding the occupancy or the stationary probability that the content data is in cache, we know that a content is cached for a duration T in a renewal interval Y. Then by renewal theory the occupancy π is the ratio $\mathbb{E}[T]/\mathbb{E}[Y] = \mu^{-1}m_R$ which completes the proof.*

Proposition 4.3 states that it is enough to compute $\mathbb{E}[Z]$ 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.

A cached content may be refreshed only after T expires, upon a cache miss. Hence the refresh rate is nothing but the miss rate in the case of a cache directly connected to the authoritative server. In the presence of intermediate caches, the refresh rate of a cache is its miss rate times the product of miss probabilities at all intermediate caches. The correctness probability of a server is the probability that a request gets the correct content, whether it was cached or not. When a cache is directly connected to the authoritative server, a cache miss ensures that the delivered content is correct whereas a cache hit may or may not provide a correct content. This will depend on the distribution of the inter-change time at the authoritative server. A thorough analysis of this metric is left for future work.

4.3 Special Distributions of Caching Durations

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

4.3.1 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 [15].

Result 4.1 (deterministic caching duration) *The expected number of hits in a renewal interval is $\mathbb{E}[Z] = M(D)$.*

Combining Result 4.1 with Proposition 4.3 yields the performance metrics. These are exactly the ones found in [15, Thm 1]. The CDF $G(t)$ of the generic inter-miss time, on the other hand, is a new result. Using $T(t) = \mathbf{1}\{t > D\}$, (3) becomes

$$G(t) = \mathbf{1}\{t > D\}\left(F(t) - \int_0^D (1 - F(t-x))dM(x)\right). \tag{6}$$

4.3.2 Exponential Distribution

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

Result 4.2 (exponential caching duration) *The expected number of hits in a renewal interval is* $\mathbb{E}[Z] = F^*(\mu)/(1 - F^*(\mu))$, *and* (4) *giving the LST of* $G(t)$ *becomes*

$$G^*(s) = \frac{F^*(s) - F^*(s+\mu)}{1 - F^*(s+\mu)}. \tag{7}$$

The result above is identical to Propositions 3.1 and 3.2 in [7]. The system considered in [7] consists of caches using expiration-based policies whose caching durations are reset at every cache hit. The DNS scenario considered in this report 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.

4.3.3 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 $1 - \alpha \exp(\mathbf{S}t)\mathbf{u}$, where α and $\mathbf{u}$ are dimension-n vectors and $\mathbf{S}$ is an $n \times n$ matrix; the ME distribution is said to be of order n. If $\mathbf{S}$ is diagonalizable,[1] then a diag.ME is obtained. The LST of its CDF is rational.

Our interest in the diag.ME is threefold. First, it covers a large set of distributions including the acyclic phase-type distributions like the generalized coxian distribution, the exponential distribution, the hypo-exponential distribution or generalized Erlang, the hyper-exponential distribution or mixture of exponentials. Second, as reported in [10], a general point process can be well fitted by a renewal process having a "phase-type distribution" such as the "mixture of exponentials". Third (and most attractively) it is analytically tractable as will become clear in Sect. 6. In brief, if inter-request times of exogenous arrivals and caching durations all follow this distribution, then any inter-miss time and any overall inter-request time in a network of caches will also follow this distribution (with other parameters), as long as an additional assumption is enforced.

The CDF of a caching duration following a diag.ME of order K can be written

$$T(t) = 1 - \sum_{k=1}^{K} b_k e^{-\mu_k t}, \quad \text{with} \quad \sum_{k=1}^{K} b_k = 1. \tag{8}$$

There is no restrictions on $\{\mu_k\}_{1 \le k \le K}$ except that $T(t)$ must be a CDF. The following then holds.

Result 4.3 (diag.ME caching duration) *The expected caching duration and the expected number of hits in a renewal interval are, respectively,*

$$\mu^{-1} = \sum_{k=1}^{K} b_k \mu_k^{-1}; \qquad \mathbb{E}[Z] = \sum_{k=1}^{K} \frac{b_k F^*(\mu_k)}{1 - F^*(\mu_k)}, \tag{9}$$

and the LST of $G(t)$ *given in* (4) *can be rewritten*

$$G^*(s) = 1 - \sum_{k=1}^{K} b_k \frac{1 - F^*(s)}{1 - F^*(s+\mu_k)}. \tag{10}$$

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

Using (9) in Proposition 4.3 yields the performance metrics.

4.4 Optimal TTL Distribution per content

This section addresses the following challenging question: which distribution optimizes the performance of a content caching policy and under which conditions? A partial answer will be provided in the following.

There are conflicting objectives when optimizing a caching policy. Caching has been introduced to limit wide-area DNS traffic and to speed up DNS lookups at clients. An efficient cache is then one that has a small miss rate, a high hit probability and yet a small occupancy (data is in cache only when needed). The counter effect is an increase in the probability for the user to obtain an outdated content. Indeed, as explained in Sect. 3, contents are refreshed only upon a cache miss. Having then a high miss rate is desirable when the content is likely to change often.

In this section, we will order distributions according to the achieved performance metrics, namely the miss rate m_R, the hit probability h_P and the occupancy π. Consider two different policies. In one policy, a content is cached for a deterministic duration D; in the other, the caching duration T has a CDF $T(t)$ such that $\mathbb{E}[T] = D$. The performance metrics vary with the distribution, the rv is then explicitely appended to the notation, e.g. $\pi(T)$.

Proposition 4.4 (optimal policy) *If inter-arrival requests at a cache have a concave CDF then the deterministic caching duration yields the most efficient caching, i.e.*

$$m_R(D) \leq m_R(T) \ , \quad h_P(D) \geq h_P(T) \ , \quad \pi(D) \leq \pi(T).$$

Proof 4.4 *Define $\phi(t) = 1 + M(t)$. We therefore have (use $\mathbb{E}[Z] = \mathbb{E}[M(T)]$ in Proposition 4.3)*

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

We will now prove that ϕ is concave. Recall that $M(t)$ is the renewal function. Differentiating twice (1) *yields*

$$\phi''(t) = m'(t) = f'(t) + \int_0^t m(t-x) f'(x) dx. \tag{11}$$

Since $m(t)$ is a positive function, it follows that $\phi(t)$ is a concave function if $F(t)$ is concave (i.e. if $f'(t) < 0$). Using now Jensen's inequality yields $\mathbb{E}[\phi(T)] \geq \phi(\mathbb{E}[T]) = \phi(D) = \mathbb{E}[\phi(D)]$ which completes the proof.

As F is a CDF, it may not be convex and the corollary of Proposition 4.4 never applies. Finding the optimal policy when F is not concave is an open problem. The simulations discussed in Sect. 7.2 suggest however that, in this latter case, the higher the coefficient of variation, the better.

The concavity of the CDF $F(t)$ of the inter-request times is not a strong condition. Jung, Berger and Balakrishnan use in [15] Pareto and Weibull (with shape less than 1) distributions to fit collected inter-request times (cf. discussion around Assumption 1 in Sect. 3). These distributions have concave CDFs. Also, it is known that long-tailed distributions having a decreasing failure rate can be well approximated by a mixture of exponentials [10], whose CDF is concave. Last, a conceptual model often used in the analysis of caches (e.g. [9, 20, 22]) is the so-called *independent reference model* (IRM). This model is equivalent to assuming that requests for a single content form a Poisson process [11]. The CDF of the (exponential) inter-arrival times is then concave.

Proposition 4.4 states that deterministic caching durations are the optimal when F is concave (Assumption 1 must hold). *This does not mean that all contents should use the same constant TTL value but rather to have a fixed value per content.* For each content which receives its own deterministic timer, the hit probability is maximized and yet the occupancy is minimized, suugesting that *the content is found in the cache mainly when needed, i.e. at requests arrivals.* The next obvious question is: which deterministic value is the optimal one? This question, already posed in [4], will be addressed now.

Since the deterministic policy is optimal only for concave F, we will only consider this case in the discussion. Ideally, the optimal deterministic value, $D^\star$, should maximize the hit probability and minimize the occupancy. For a deterministic caching duration, these have the following expressions (combine Result 4.1 and Proposition 4.3):

$$h_P(D) = 1 - \frac{1}{1+M(D)}, \qquad m_R(D) = \frac{\lambda}{1+M(D)}, \tag{12}$$

$$\pi(D) = \frac{\lambda D}{1+M(D)} \tag{13}$$

For concave F, the renewal function $M(D)$ is also concave (and increasing) (cf. (11)). It is clear from (12) that the hit probability $h_P(D)$ is concave increasing (and the miss rate $m_R(D)$ convex decreasing).

Introduce now the function $g(D) = 1 + M(D) - Dm(D)$. The derivative of $\pi(D)$ w.r.t. D yields $\pi'(D) = \frac{\lambda\, g(D)}{(1+M(D))^2}$. Given that $g(0) = 1$ and $g'(D) = -Dm'(D) \geq 0$ for any $D \geq 0$ (recall that $m'(D) < 0$ for concave F), the function g is thus always positive and so is π'. Hence, the occupancy is an increasing function of the caching duration. It is therefore not possible to maximize $h_P(D)$ while minimizing $\pi(D)$, as both increase with the caching duration D.

We believe that having a high hit probability supersedes the desire of having a low occupancy. However, the miss rate should not be minimized (its minimum is 0 when $D \to \infty$) as it directly relates to the correctness of the cached content. Cache misses must occur in order to update the content.

The proper thing to do in such a case is to solve a constrained optimization problem, looking for instance to maximize the hit probability subject to a maximal occupancy $\pi_{\max}$ (for cache size issues) and/or a minimal miss rate $m_{R,\min}$ (for correctness issues). Given the monotonicity of h_P, m_R and π (for concave F), the solution is readily found as

$$D^\star = \min\{\arg \pi_{\max}, \arg m_{R,\min}\}.$$

The maximal occupancy $\pi_{\max}$ for a given content can be for instance the fraction of the cache size that is proportional to the content's popularity.

5 Applicability in Other Contexts

5.1 Single Traditional DNS Cache

The modern DNS cache analyzed in Sect. 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 Sect. 4 apply in the case of a single traditional DNS cache, as long as Assumptions 1-2 hold. Note that the model developed in [15] provides *approximate* results for a single traditional DNS cache, as

the authors consider a deterministic caching duration that is set to the maximum value among all those observed in the responses. The results of [15] are exact for a single modern DNS cache that chooses a deterministic caching duration. Our model yields *exact* results for a traditional cache *and* a modern cache, regardless of the distribution chosen for the caching durations.

5.2 Single Cache with Limited Buffer

In a cache with limited buffer capacity, there is a constraint on the number of contents that can be cached simultaneously. In caches where all contents are equally sized, like in DNS caches, the buffer size can be expressed as the maximum number of content that can be cached, let B be this number. The analysis of Sect. 4 is useful in this case as it helps setting the caching durations in such a way that the capacity constraint is satisfied.

Let $\mathcal{R}$ be the total number of contents that could be cached and let r be a given content ($r = 1, \ldots, \mathcal{R}$). All the notation relative to content r will have an additional subscript r. The capacity constraint is then written as follows

$$\sum_{r=1}^{\mathcal{R}} \pi_r \leq B \qquad \Leftrightarrow \qquad \sum_{r=1}^{\mathcal{R}} \frac{\lambda_r/\mu_r}{1+\mathbb{E}[Z_r]} \leq B. \tag{14}$$

Note that the expected number of hits $\mathbb{E}[Z_r]$ is a function of the expected caching duration μ_r^{-1}. Should the same caching policy be applied to all contents, i.e., $\mu_r = \mu$, then μ can be found as the solution of a fixed point equation

$$\mu = \frac{1}{B} \sum_{r=1}^{\mathcal{R}} \frac{\lambda_r}{1+\mathbb{E}[Z_r]}. \tag{15}$$

6 Analysis of a Network of Caches

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*. To analyze it, one additionally needs to consider the network topology. The notation relative to cache c will have an extra subscript c. Assumptions 1-4 are enforced throughout this section. Requests for a given content may only flow over a tree network and exogenous arrivals are independent so that Assumption 3 holds. In the following we consider the particular case of linear networks for which exact results can be derived (cf. Sect. 6.1). We will move next to the general tree network case for which approximate results can be derived by enforcing an additional assumption (cf. Sect. 6.2). Last, we focus on the particular case where caching durations and exogenous inter-request times follow a diag.ME distribution (cf. Sect. 6.3). Results for this last case are interesting as the diag.ME distribution will be preserved inside the network.

6.1 Linear Networks: Exact Results

Consider the linear network depicted in Fig. 2. There are C caches and the disk of the authoritative server (the rightmost cache is the one of the authoritative server). By Assumption 1, the overall request process at cache 1 is a renewal process. By Proposition 4.2, the miss process at cache 1 (which is nothing but the request process at cache 2) is also a renewal process. Hence, all processes in this linear network of caches are renewal processes. The performance metrics at each cache are derived using Proposition 4.3.

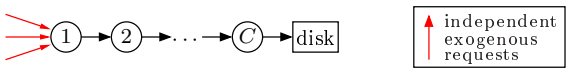


Figure 2: A linear network with C caches.

6.2 Tree Networks: a Recursive Procedure

The aggregation of several renewal processes in *not* a renewal process. However, it is mandatory to have a renewal process for Proposition 4.1 to hold at any high-level cache inside the network. Similarly to [7], we overtake this limitation by proceeding as if we do have a renewal process, and then assess the robustness of the model against situations where this is not the case. The approximate results obtained are strikingly accurate as will be seen later in Sect. 7.2. In the rest of the report, the following assumption will be enforced.

Assumption 5 (aggregation) *The overall request arrival process at each cache is a renewal process.*

A direct consequence of Assumption 5 is that the miss process at each cache is a renewal process thanks to Proposition 4.1. Propositions 4.2 and 4.3 are also valid at any cache. For the case of a single cache, the CDF of the inter-miss time at a cache, namely $G(t)$, is expressed as a function of the CDF of the inter-request time, namely $F(t)$; see (3).

In the case of a network, one needs to consider at a cache c the inter-request time of the *aggregate* process arriving at cache c. Let $H_c(t)$ be its CDF. Equation (3) provides the CDF of the inter-miss time at cache c, denoted by $G_c(t)$, after replacing $F(t)$ with $H_c(t)$ and by using the renewal function associated with the aggregate request process, say $M_c(t)$, in (2). To explicitly write this equation for the case of a network of caches, additional notation is needed.

The set of children of cache c is $\mathcal{C}(c)$ with $C = |\mathcal{C}(c)|$. The rate of *exogenous* requests (if any) at cache c is λ_c; the CDF of inter-exogenous request times is $F_c(t)$. There are $C+1$ request processes at cache c. Their aggregation has a rate

$$\Lambda_c = \lambda_c + \sum_{i \in \mathcal{C}(c)} m_{R,i}. \tag{16}$$

The C miss processes at the children of c and the exogenous request process at cache c are all independent. Thereby, the result derived by Lawrance in [17, Eq. (4.1)] applies. By Assumption 5, the aggregate request process at cache c is a renewal process and the CCDF of the inter-request time is

$$\begin{aligned} \bar{H}_c(t) \;&=\; \frac{\lambda_c}{\Lambda_c}\bar{F}_c(t) \prod_{i \in \mathcal{C}(c)} m_{R,i} \int_t^\infty \bar{G}_i(u)du \\ &+ \sum_{i \in \mathcal{C}(c)} \frac{m_{R,i}}{\Lambda_c}\bar{G}_i(t)\lambda_c \int_t^\infty \bar{F}_c(u)du \prod_{\substack{j \in \mathcal{C}(c) \\ j \neq i}} m_{R,j} \int_t^\infty \bar{G}_j(u)du. \end{aligned} \tag{17}$$

Equation (5) becomes

$$G_c(t) = H_c(t) - \int_0^t (1 - H_c(t-x))\bar{T}_c(x)dM_c(x) \tag{18}$$

with $\bar{T}_c(t)$ the CCDF of the caching duration at cache c and $M_c(t)$ the renewal function associated with the aggregate request process at the same cache. Equations (17)-(18) provide a recursive

procedure for calculating the CDFs $H_c(t)$ and $G_c(t)$ at each cache c of a tree network. Numerical procedures such as Romberg's method or other techniques for computing (17)-(18) recursively can be found in [23]. We consider next a special case in which closed-form expressions for $H_c(t)$ and $G_c(t)$ can be found.

6.3 Closed-Form Results with diag.ME RVs

In this section, we consider a tree 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

$$F_c(t) = 1 - \sum_{j=1}^{J_c} a_{c,j} e^{-\lambda_{c,j} t}, \; \bar{T}_c(t) = \sum_{k=1}^{K_c} b_{c,k} e^{-\mu_{c,k} t}, \tag{19}$$

for $t > 0$. J_c and K_c are the respective orders of the diag.ME distributions. We are now in position to prove an interesting property that is another main contribution of this work. This property is the self-preservation of the diag.ME distribution across a tree network as stated in what follows.

Proposition 6.1 (diag.ME preservation) *Under Assumptions 1-5 and as long as* (19) *is satisfied at each cache c of a tree network, miss processes and aggregate requests are all renewal processes whose inter-event time follows a diag.ME distribution (parameters are in the proof).*

Proof 6.1 *The proof rests on three arguments: (i) the miss process at each of the lowest-level caches checks Proposition 6.1; (ii) the aggregate request process and (iii) the miss process at each of the next higher-level caches verify Proposition 6.1. Arguments (ii) and (iii) will be used repeatedly until all caches in the network are covered. By Proposition 4.1 and Assumption 5, the processes at hand are renewal processes. We focus then on the distribution of the inter-event time.*

Argument (i): the miss process at a lowest-level cache. Let c be such a lowest-level cache, it corresponds to a leave in a tree. The CDF of the inter-request time is given by (19)*. The renewal equation* (1) *can be written as follows*

$$M_c(t) = F_c(t) + \int_0^t \sum_{j=1}^{J_c} a_{c,j} \lambda_{c,j} e^{-\lambda_{c,j}(t-x)} M_c(x) dx. \tag{20}$$

The solution of (20) *is given in [19, Sect. 2.2.1.19] which we can differentiate to find*

$$dM_c(t) \;\; = \;\; \sum_{j=1}^{J_c} \gamma_{c,j} e^{-\theta_{c,j} t} dt \tag{21}$$

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

$$0 = 1 - \sum_{j=1}^{J_c} \frac{a_{c,j} \lambda_{c,j}}{\lambda_{c,j} - z}, \tag{22}$$

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

$$\Big\{ 0 = 1 + \sum_{j=1}^{J_c} \frac{\gamma_{c,j}}{\theta_{c,j} - \lambda_{c,n}}, \qquad 1 \le n \le J_c. \tag{23}$$

Combining now (19) *and* (21)*, we can apply Proposition 4.2 to rewrite* (5) *as follows*

$$G_c(t) = 1 - \sum_{j=1}^{J_c} a_{c,j} \left(1 + \sum_{k=1}^{K_c} \sum_{i=1}^{J_c} \frac{b_{c,k}\gamma_{c,i}}{\theta_{c,i} + \mu_{c,k} - \lambda_{c,j}} \right) e^{-\lambda_{c,j}t} - \sum_{k=1}^{K_c} \sum_{i=1}^{J_c} \left(\sum_{j=1}^{J_c} \frac{(-a_{c,j})b_{c,k}\gamma_i}{\theta_{c,i} + \mu_{c,k} - \lambda_{c,j}} \right) e^{-(\theta_{c,i}+\mu_{c,k})t}. \quad (24)$$

Clearly, the inter-miss time at a lowest-level cache follows a diag.ME distribution, whose order is $J_c(K_c + 1)$ *which is the number of exponentials in* (24)*.*

Argument (*ii*)*: the aggregate request process at a next higher-level cache. The CCDF of the inter-request time at this intermediate cache* c *is given in* (17)*, where* $F_c(t)$ *is relative to the exogenous request process and* $G_i(t)$ *is relative to the* i*th cache in* $\mathcal{C}(c)$*, the set of children of cache* c*. Recall that* $C = |\mathcal{C}(c)|$*. To ease the derivation of* $H_c(t)$*, we rewrite* $F_c(t)$ (19) *and* (24) *with a new/modified notation* ($t > 0$)

$$F_c(t) = 1 - \sum_{l_0=1}^{\mathcal{L}_0} a_{0,l_0} e^{-\lambda_{0,l_0}t}, \quad G_i(t) = 1 - \sum_{l_i=1}^{\mathcal{L}_i} a_{i,l_i} e^{-\lambda_{i,l_i}t}.$$

The exogenous request rate is denoted $r_0 = \sum_{l_0=1}^{\mathcal{L}_0} a_{0,l_0}\lambda_{0,l_0}$*. The miss rate at the* i*th cache in* $\mathcal{C}(c)$ *is denoted* r_i*. The overall request rate at cache* c *becomes* $\Lambda_c = \sum_{i=0}^{C} r_i$ *(see* (16)*). After tedious calculations,* (17) *can be rewritten*

$$\bar{H}_c(t) = \frac{\prod_{i=0}^{C} r_i}{\Lambda_c} \sum_{l_0=1}^{\mathcal{L}_0} \sum_{l_1=1}^{\mathcal{L}_1} \cdots \sum_{l_C=1}^{\mathcal{L}_C} \sum_{i=0}^{C} \lambda_{i,l_i} \left(\prod_{j=0}^{C} \frac{a_{j,l_j}}{\lambda_{j,l_j}} \right) \exp\left(- \left(\sum_{j=0}^{C} \lambda_{j,l_j} \right) t \right). \quad (25)$$

The inter-request time at the intermediate cache c *follows a diag.ME distribution of order* $\prod_{i=0}^{C} \mathcal{L}_i$*.*

Argument (*iii*)*: the miss process at a next higher-level cache. Argument* (*i*) *can be repeated here by carefully replacing the exogenous request process with the aggregate request process discussed in Argument* (*ii*)*. We can conclude that it is enough to have the caching duration at a cache and the inter-request time at the same cache follow a diag.ME distribution for the inter-miss process at this cache to follow a diag.ME distribution. This completes the proof.*

The performance metrics can be found at each cache by using Result 4.3 and Proposition 4.3. 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 4.3 at each higher-level cache.

Sections 6.2 and 6.3 provide approximate results as Assumption 5 is not true. The robustness of our model is tested in Sect. 7.2.

7 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 Sect. 4 to results derived from a real DNS cache trace. The case of a network of caches is addressed next, where the objective is to validate Assumption 5.

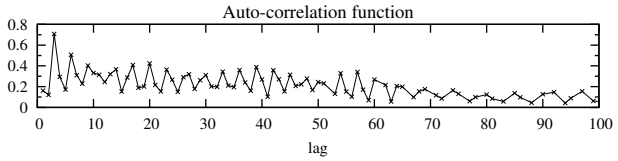
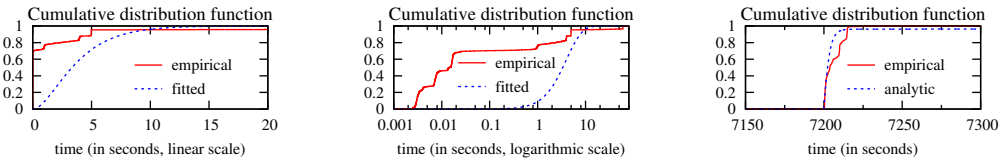


Figure 3: Content ranked 6th: correlated inter-requests.



(a) inter-request times (b) inter-request times (c) inter-miss times

Figure 4: Comparison of CDFs for content ranked 6th.

7.1 Using a Real Trace (Single Cache)

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. The trace contains information about 2313984 resource records requested by a total of 2147 users. Processing the trace provides, for each resource record (or content):

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).

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, requests are time-varying (week day/week-end, day/night) and clearly dependent as illustrated in Fig. 3 for one of the contents (cf. lags 3 and 6). Therefore, Assumption 1 (renewal request process) is not met. *Testing our model using this trace will give insights on its robustness* since the main assumptions used in the single cache analysis are not met in this trace. Third, 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. The TTLs found in the final response packets vary from 1 to the initial TTL advocated by authoritative servers; this emphasizes the pertinence of our models as caches at the user side are given non-deterministic TTLs.

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 randomly picked one resource record out of the most requested among users. The caching duration of the chosen content (ranked 6th) turns out to be deterministic and equal to 2 hours (value provided directly by five authoritative servers). We used the KPC-Toolbox [5] to find the Markovian Arrival Process (MAP) that best fits the inter-request times

Table 2: Performance Metrics and Relative Errors (Rank 6)

Metric	Trace	Model	Rel. err. (%)
miss rate	0.00013876	0.00013749	0.920
hit probability	0.99943	0.99941	0.002
occupancy	0.99914	0.98995	0.920

Table 3: Analytic Performance Metrics and their Relative Errors (in Percentage) at Representative Caches ($\lambda_1 = 1.57$ requests/s, $\lambda_2 = 0.87$ requests/s, $\lambda_3 = 1.37$ requests/s, $\lambda_4 = 0.68$ 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.	
1	miss rate	**0.49479**	0.00921	0.49906	0.00649	0.50039	0.08715	0.50235	0.07702	↗
	hit probability	**0.43275**	0.03832	0.42785	0.02724	0.42632	0.00660	0.42408	0.00065	↘
	occupancy	**0.35786**	0.04466	0.36094	0.04712	0.36191	0.03360	0.36333	0.02360	↗
5	miss rate	0.56708	1.1214	0.52673	0.08478	0.51681	0.10264	**0.51073**	0.00132	↘
	hit probability	0.41611	1.4561	0.46389	0.18679	0.47589	0.1514	**0.48412**	0.10321	↗
	occupancy	0.58169	1.146	0.54023	0.04850	0.53005	0.06307	**0.52379**	0.04179	↘
7	miss rate	0.52928	5.0614	0.48234	0.23668	0.46971	0.06873	**0.46045**	0.00650	↘
	hit probability	0.51789	4.536	0.52049	0.25253	0.52361	0.1067	**0.52731**	0.07069	↗
	occupancy	0.67667	5.0986	0.61667	0.19648	0.60051	0.02771	**0.58866**	0.03662	↘

X of the aggregated arrival process (generated by 145 different users). 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 is 128. The moments of the empirical inter-request time (as computed by the tool) are: mean = 4.1614, variance = 4476.9, skewness 83.8809, kurtosis 7973.3. For completeness, we depict in Figs. 4a-4b the empirical and fitted distributions of the inter-request times in linear and logarithmic scale respectively.

Taking as input the fitted distribution and the TTL value, we use the findings of Sect. 4.3.1 to obtain the performance metrics of the cache relative to the content ranked 6th (cf. Table 2) and the CDF of the inter-miss times (cf. Fig. 4c). To determine the CDF (6), 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, the larger τ and the smaller Δ, the smaller the numerical error but also the larger the computational cost. We set $\tau = 720000$ (100 times the maximum between the mean inter-request time and the TTL) and $\Delta = 0.1$.

The analytic results are compared to those computed from the trace. Table 2 reports *negligible values of the relative errors on the performance metrics. Proposition 4.3 appears to be applicable even if Assumption 1 is not met.* In fact, we believe that it is enough to have *stationary and ergodic* point processes as requests for Proposition 4.3 to apply; cf. [2, Eq. (1.3.2), p. 21]. Lawrence's theorem [17, Eq. (4.1)] can then be replaced with [2, Eq. (1.4.6), p. 35].

As for the miss process, Fig. 4c is clear: *our model accurately estimates the CDF of the inter-miss time. Proposition 4.2 appears to be applicable even if Assumption 1 is not met.* This section suggests that *our single cache model is robust.*

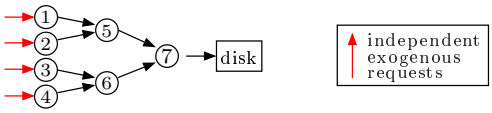


Figure 5: A binary tree with 7 caches.

7.2 Validating Assumption 5

We now proceed to evaluating the robustness of our model of a network of caches. To this end, we resort to performing event-driven simulations. It is worth recalling that with exponentially distributed caching durations our model coincides with the one developed in [7] to study caches that reset the caching durations at each hit. In [7], Assumption 5 is also used; the authors evaluate the robustness of 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 Assumption 5. The same applies to our model when caching durations are exponentially distributed.

We consider a tree consisting of 7 caches as shown in Fig. 5. This tree represents well the hierarchy found in DNS: cache 7 is that of the authoritative server, caches 5 and 6 are typically those of ISP's DNS servers, and caches 1-4 are found at the client side (ADSL modem, laptop, etc.). To capture the fact that users have interleaving activity and inactivity periods, requests for all contents are assumed to form a Markov-Modulated Poisson Process (MMPP). In other words, requests for a single content form an Interrupted Poisson Process (IPP). As a consequence, Assumption 5 is not satisfied at caches 5, 6 and 7 since each component (miss process) of their overall request process is not a Poisson process.

In each performed simulation, we consider a single content whose requests at each bottom-level cache form an IPP. The (exogenous) request rate at cache i is $\lambda_i \in [0.5, 20]$ for $i \in \{1, 2, 3, 4\}$.

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.

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 Sect. 6.2. We have implemented a MATLAB numerical solver that determines the CDFs in the network (using (17)-(18)) and then the metrics of interest at each cache (using Proposition 4.3 where $\mathbb{E}[Z_c] = L_c(\infty)$). The numerical error comes from the integral computation used in (17)-(18) (e.g., the integrals over infinite ranges). Again, we use Riemann's sum and, for simplicty, 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.

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 3 (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 conclude that *using Assumption 5 is not a limitation* and that *our model is very robust* to violations of this Assumption.

7.3 Optimal Caching Policy in a Network

According to Sect. 4.4, if the CDF of inter-request times at a cache is concave, then the best caching policy is to cache a content for a deterministic duration. If exogenous request processes satisfy this condition, it will not be the case of the aggregate request process reaching a higher-level cache.

Consider again the simulations presented in Sect. 7.2. Table 3 reports in columns 3, 5, 7, and 9 the analytic values of the performance metrics obtained at caches 1, 5 and 7 (one cache at each level) of the synthetic network of Fig. 5. The trend observed on these metrics as the distribution changes from the least variable (i.e., the deterministic) to the most variable (i.e., the hyper-exponential) is shown in column 11.

The optimal values of the performance metrics are in bold fonts in Table 3. The best distribution at bottom-level caches (e.g., cache 1) is the deterministic one. This is predicted by Proposition 4.4 which applies here as the inter-request time of an IPP has a concave CDF. The trend on each of the metrics is inverted at higher-level caches. The deterministic policy achieves then the worst performance. The more variable a distribution, the better the performance metrics. The inter-request time at higher-level caches no longer has a concave CDF. Recall that these observations are for each content individually. The parameters of a given distribution will vary from a content to another according to the popularity.

The above trends are observed when all the caches in a tree use the same distribution. Since we have established that for concave CDF (the case of IPP requests) the deterministic distribution is the best, we repeated the simulations described earlier with the exception of having deterministic TTLs at all bottom-level caches. We observed the same trends for the same values of λ_i for $i \in \{1, 2, 3, 4\}$ as in Table 3 and also for another set of values that is $\lambda_1 = 0.052$ requests/s, $\lambda_2 = 0.061$ requests/s, $\lambda_3 = 0.091$ requests/s, $\lambda_4 = 0.078$ requests/s.

Our study suggests that for better performance, *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).*

8 Conclusions

The analytic models introduced in this report proved to be very useful 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 the performance metrics and the CDF of the miss process remarkably well. The main approximation used in our network of caches model has been validated through simulations. We have addressed the problem of the optimal caching duration and found that if inter-request times have a concave CDF, then the deterministic policy is the best. For non-concave CDF, our numerical analysis suggests that more variable distributions are better. We plan to pursue the validation of our model using the real traces collected.

9 Acknowledgments

The authors would like to thank Francis Montagnac (IT staff at Inria, Sophia Antipolis) for collecting the DNS traces. The authors are deeply grateful to Fabrice Huet (Associate professor, University Nice Sophia Antipolis) for his help in processing the large amount of data collected.

References

[1] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman. A survey of information-centric networking. *IEEE Communications Magazine*, 50(7):26–36, July 2012.

[2] F. Baccelli and P. Brémaud. *Elements of Queueing Theory, Palm Martingale calculus and Stochastic recurrences*. Springer, Berlin, 2nd edition, 2003.

[3] R. J. Bayardo, R. Agrawal, D. Gruhl, and A. Somani. YouServ: a web-hosting and content sharing tool for the masses. In *Proc. ACM WWW'02*, pages 345–354, New York, USA, 2002.

[4] T. Callahan, M. Allman, and M. Rabinovich. On modern DNS behavior and properties. *ACM SIGCOMM Comp. Comm. Rev.*, 43(3):7–15, 2013.

[5] G. Casale, E. Zhang, and E. Smirni. KPC-Toolbox: Simple yet effective trace fitting using Markovian Arrival Processes. In *Proc. 5th Intl. Conf. on the Quantitative Evaluation of SysTems (QEST'08)*, 2008.

[6] H. Che, Y. Tung, and Z. Wang. Hierarchical Web caching systems: modeling, design and experimental results. *IEEE J. on Selected Areas in Communications*, 20(7):1305–1314, 2002.

[7] N. Choungmo Fofack, P. Nain, G. Neglia, and D. Towsley. Analysis of TTL-based cache networks. In *Proc. ACM ValueTools'12*, Cargèse, France, Oct. 2012.

[8] D. R. Cox. *Théorie du Renouvellement*. Monographies DUNOD, Paris, 1966.

[9] A. Dan and D. Towsley. An approximate analysis of the LRU and FIFO buffer replacement schemes. In *Proc. ACM SIGMETRICS'90*, pages 143–152, Boulder, CO, USA, May 1990.

[10] A. Feldmann and W. Whitt. Fitting mixtures of exponentials to long-tail distributions to analyze network performance models. In *Proc. IEEE INFOCOM'97*, Kobe, Japan, Apr. 1997.

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

[12] Y. T. Hou, J. Pan, B. Li, and S. Panwar. On expiration-based hierarchical caching systems. *IEEE J. on Selected Areas in Communications*, 22(1), 2004.

[13] Y. T. Hou, J. Pan, K. Sohraby, and S. X. Shen. Coping miss synchronization in hierarchical caching systems with nonlinear TTL functions. In *Proc. IEEE ICC'04*, pages 2194–2198, 2004.

[14] C. Jackson, A. Barth, A. Bortz, W. Shao, and D. Boneh. Protecting browsers from DNS rebinding attacks. *ACM Trans. Web*, 3(1), 2009.

[15] J. Jung, A. W. Berger, and H. Balakrishnan. Modeling TTL-based Internet caches. In *Proc. IEEE INFOCOM'03*, San Francisco, CA, USA, Mar. 2003.

[16] J. Jung, E. Sit, H. Balakrishnan, and R. Morris. DNS performance and the effectiveness of caching. In *Proc. ACM SIGCOMM Workshop on Internet Measurement (IMW '01)*, New York, NY, USA, Nov. 2001.

[17] A. T. Lawrance. Dependency of intervals between events in superposition processes. *J. of the Royal Statistical Society, Series B (Methodological)*, 35(2):306–315, 1973.

[18] J. Pang, A. Akella, A. Shaikh, B. Krishnamurthy, and S. Seshan. On the responsiveness of DNS-based network control. In *Proc. IMC*, Taormina, Italy, 2004.

[19] A. D. Polyanin and A. V. Manzhirov. *Handbook of Integral Equations*. CRC Press, 1st edition, 1998.

[20] E. J. Rosensweig, J. Kurose, and D. Towsley. Approximate models for general cache networks. In *Proc. IEEE INFOCOM'10*, San Diego, USA, 2010.

[21] S. Saroiu, K. P. Gummadi, R. J. Dunn, S. D. Gribble, and M. Levy. An analysis of Internet content delivery systems. *SIGOPS Operating System Review*, 36:315–327, 2002.

[22] A. Simonian, M. Gallo, B. Kauffmann, L. Muscariello, and C. Tanguy. Performance of the random replacement policy for networks of caches. In *Proc. ACM SIGMETRICS/PERFORMANCE'12*, pages 395–396, London, England, UK, June 2012.

[23] M. Tortorella. Numerical solutions of renewal-type integral equations. *INFORMS J. on Computing*, 17:73–96, 2005.

Contents

1 Introduction **3**

2 Related Works **4**
2.1 TTL rule and caching durations 4
2.2 State-of-the-art in modeling DNS caches 4

3 Scenario, Definitions, Assumptions **5**
3.1 Considered Scenario 5
3.2 Metrics and Properties of a Cache 6
3.3 Processes at Hand 6

4 Analysis of a Single Cache **7**
4.1 The Model and its Analysis 8
4.2 Performance Metrics 9
4.3 Special Distributions of Caching Durations 10
4.3.1 Deterministic Distribution 10
4.3.2 Exponential Distribution 11
4.3.3 Diagonal Matrix-Exponential Distribution 11
4.4 Optimal TTL Distribution per content 12

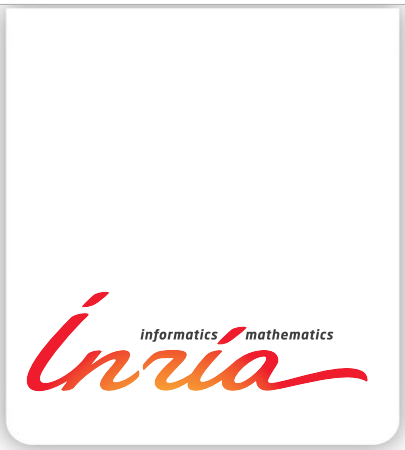
5 Applicability in Other Contexts **13**
5.1 Single Traditional DNS Cache 13
5.2 Single Cache with Limited Buffer 14

6 Analysis of a Network of Caches **14**
6.1 Linear Networks: Exact Results 14
6.2 Tree Networks: a Recursive Procedure 15
6.3 Closed-Form Results with diag.ME RVs 16

7 Validation, Numerical Results **17**
7.1 Using a Real Trace (Single Cache) 18
7.2 Validating Assumption 5 20
7.3 Optimal Caching Policy in a Network 21

8 Conclusions **21**

9 Acknowledgments **22**



RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399