Performance evaluation of hierarchical TTL-based cache networks

Nicaise Choungmo Fofack, Philippe Nain, Giovanni Neglia, Don Towsley

Performance Evaluation of Hierarchical TTL-based Cache Networks[☆]

Nicaise Choungmo Fofack[a,b], Philippe Nain[b], Giovanni Neglia[b], Don Towsley[c]

[a] *University of Nice Sophia Antipolis, 28 Avenue Valrose, 06100 Nice, France*

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

[c] *University of Massachusetts at Amherst, 181 Governors Dr Amherst, MA 01003-9264, USA*

Abstract

There has been considerable research on the performance analysis of *on-demand* caching replacement policies like Least-Recently-Used (LRU), First-In-First-Out (FIFO) or Random (RND). Much progress has been made on the analysis of a single cache running these algorithms. However it has been almost impossible to extend the results to networks of caches. In this paper, we introduce a Time-To-Live (TTL) based caching model, that assigns a timer to each content stored in the cache and redraws it every time the content is requested (at each hit/miss). We derive the performance metrics (hit/miss ratio and rate, occupancy) of a TTL-based cache in isolation fed by stationary and ergodic request processes with general TTL distributions. Moreover we propose an iterative procedure to analyze TTL-based cache networks under the assumptions that requests are described by *renewal processes* (that generalize Poisson processes or the standard IRM assumption). We validate our theoretical findings through event-driven and Monte-Carlo simulations based on the Fourier Amplitude Sensitivity Test to explore the space of the input parameters. We observe that our analytic model predicts remarkably well all metrics of interest with relative errors smaller than 1%.

Keywords: Performance analysis, Content-centric networks, Time-To-Live, Cache replacement policy, Renewal theory, Markov model

[☆] This manuscript is an extended version of [7].

Email addresses: nicaise.choungmo_fofack@inria.fr (Nicaise Choungmo Fofack), philippe.nain@inria.fr (Philippe Nain), giovanni.neglia@inria.fr (Giovanni Neglia), towsley@cs.umass.edu (Don Towsley)

1. Introduction

Caches are widely used in networks and distributed systems to improve their performance. They are integral components of the World Wide Web [10], the Domain Name System (DNS) [32], and Content Distribution Networks (CDNs) [44]. More recently there has been a growing emphasis on Information-Centric Networking (ICN) [1] architectures—like the Content-Centric Network (CCN) [28]—which support host-to-content interactions as the common case. Many of these *content networks* give rise to hierarchical (or more general) cache topologies. The design, the configuration and the analysis of these cache systems pose significant challenges.

An abundant literature exists on the performance (e.g. hit probability, search cost) of a single cache running the First-In-First-Out (FIFO) or Random (RND) replacement policies (see [24] for independent and identically distributed or i.i.d. requests), the Least-Recently-Used (LRU) replacement policy or, its companion, the Move-to-Front (MTF) policy (see [4, 5, 6, 15, 19, 22, 27, 29, 36] for i.i.d. requests and [13, 31, 30] for correlated requests). With few exceptions, exact models of caches in isolation are computationally intractable, resulting in the reliance on approximations [15, 29]. Networks of caches are significantly more difficult to analyze and no exact solution has been obtained so far for even the simple network of two LRU (or FIFO, RND) caches in tandem. Approximations have been proposed for star networks of LRU and RND caches by [10, 23] and [45] respectively. [43] is one of the first modeling attempt to approximate the performance of a general network of LRU caches. However, theses approximate models suffer from inaccuracies as reported in [43] where the relative error can reach 16%. Despite the increasing interest in ICN architectures, previous work has mainly focused on global architecture design. An exception is [9], whose authors develop approximations to calculate the stationary throughput in a CCN network of LRU caches modeling the interplay of chunk-level caching and a receiver-driven transport protocol. In the literature, the *5-Minute Rule* by [25] is probably one of the first paper to describe a Time-To-Live (TTL) based algorithm to manage data in computer memories. [33] considers a *single* TTL-based cache fed by i.i.d. requests to study the timer-based expiration policy of DNS caches in isolation. According to the RFC 6195, each missed resource record is marked with a timeout which indicates the maximum duration the record can be stored in the DNS cache. The timeouts are initialized only by an authoritative DNS server and an eventual hit on a local DNS cache does not change the value of the remaining timeout. Therefore, DNS caches are different from our TTL-based systems. [33] obtains the hit rate of a single DNS cache for a constant TTL via the solution of a renewal equation.

In this paper, we focus on a class of caches we introduced in our previous work [7] and we refer to as Time-To-Live (TTL)-based caches. Here TTLs are not used to guarantee the consistency of dynamic contents (as it is the case of [3, 14, 33, 46]), but to implement an eviction policy that decides which contents have to be kept in the cache. Briefly, when an uncached data item is brought back into the cache due to a cache miss, a local TTL is set and further redrawn at every cache hit.[1] The TTL value can be different for different data, but also for the same data item at different caches. All requests to that data item before the expiration of the TTL are successful (cache hits); the first request for that data item to arrive after the TTL expiration yields a cache miss. In this latter case, the cache may forward the request to a higher-level cache, if any, or to the server. When located, the data item is routed on the reverse-path and a copy is placed in each cache along the path (as in CCN [28]). This paper makes the case that TTL policies are interesting alternatives to policies such as LRU or RND for three reasons. First, a TTL policy is more configurable and in particular can mimic the behavior of other replacement policies through a proper choice of parameters (see Section 7.2). Second, while LRU or RND cache networks have defied accurate analysis, networks of TTL-based caches are simpler to study (as we show in Sections 3 and 4). Finally, the TTL-based model appears as a unified framework for the performance analysis of heterogeneous cache networks where the caches may run different replacement policies. Precisely, we develop a set of building blocks for the analysis of hierarchical TTL-based cache networks, where (*i*) exogenous requests at different caches are modeled as independent renewal processes, and (*ii*) independent TTL values are drawn at each cache from arbitrary distributions.
The building blocks are:

1. a model of a *single content* TTL cache fed by a renewal request stream (or a more general stationary request process),
2. a renewal process approximation of the superposition of independent renewal processes.

The first block forms the basis to evaluate the performance metrics and to describe the output sequence of requests (the miss process) of a cache. Meanwhile, the second block is used to characterize the resulting process of the superposition of several independent streams of requests consisting of exogenous requests from users and/or missed requests from other caches if any.

[1] This is then different from the timeouts of DNS caches; and thus, the TTL-based cache model presented here is different from the one of [33] since the TTLs are reset at every cache hit and not initialized by a central entity.

These blocks are applied to assess the performance metrics of hierarchical TTL-based cache networks. We then show how the computational cost of our approach simplifies when TTLs and the inter-arrival times of the exogenous request streams at every cache are Matrix-Exponentially Distributed (MED). We refer to this case in short as a *MED cache network*. The class of Matrix-Exponential distributions coincides with the class of distributions having a rational Laplace-Stieltjes Transform that can be used to fit properties of general processes [17, 26, 34, 41]. Event-driven and Monte-Carlo simulations on instances of MED cache networks reveal that the relative errors between the simulated networks and our model predictions are extremely small and less than 10^{-2} for all metrics of interest.

The contributions of the paper are:

- the proposal of TTL-based replacement policies for content-routers of ICN architectures,
- an analytic tool to assess the performance of hierarchical TTL-based cache networks.

This paper extends our previous work [7] as follows.
The performance metrics of single cache derived in [7] when requests were described by renewal processes are now extended (see Section 3, Propositions 3.1 and 3.2) to the case when requests are described by stationary and ergodic processes. We also provide physical and/or probabilistic interpretations of several quantities. This paper clarifies the scope of application of our theoretic results and points out our contribution with respect to several recent papers [1, 10, 11] devoted to the analysis of classical replacement policies such as LRU or RND as special case of TTL-based caches. A new result on the optimal TTL configuration of caches in isolation is added in Section 3, Proposition 3.4 and the proof is provided in Appendix. The recursive procedure presented in [7] for class $\mathcal{N}$ networks (i.e. caterpillar networks of exponentially distributed TTL-based caches fed by hyper-exponential renewal processes) is generalized by the class of MED cache networks in three orthogonal directions: (*i*) network topology considered is now an arbitrary tree of caches, (*ii*) requests are now described by a versatile class of renewal processes where inter-arrival times of requests are matrix-exponentially distributed , and (*iii*) the TTLs are drawn from matrix-exponential distribution.
The model validation in Section 5 provides additional insights on the accuracy of our assumptions and approximations, and also the efficiency of our approach in terms of computational time under various conditions. Precisely, we add results for larger networks with up to forty caches, different network topologies, two different workload models (requests described by Poisson and Interrupted

Poisson processes), hyper-exponential and hypo-exponential TTL distributions.

The paper is organized as follows. In Section 2, we introduce the notation and the model assumptions. Section 3 contains our model of a single TTL-based cache and provides the exact characterization of the performance metrics and the miss process. We describe in Section 4 a general procedure to study any hierarchical TTL-based cache network. The key point in this section is how we model the combined exogenous and miss requests streams as a renewal point process thanks to a result from [40, Eq.(4.1)], [2, Eq.(1.4.6)] regarding the computation of the marginal inter-arrival distribution for a superposition of independent renewal processes. A simplified procedure is then derived for MED cache networks. The accuracy of the general and of the simplified procedures is evaluated in Section 5 and a discussion of the computational complexity of our analytic approach can be found in Section 6. Section 7 discusses how our TTL-based model can be implemented under finite capacity constraints, and how the TTL policy can mimic different policies like LRU or RND. Conclusions are found in Section 8.

2. Single TTL cache: model and notation

For the sake of readability, we first introduce our main assumptions and our notation for the simple architecture of a single TTL-based cache and a server connected in tandem, as shown in Figure 1. The terminology and the formalism introduced here will be extended later to hierarchical TTL-based cache networks (see Section 4). From now on the words "cache" and "node" will be used interchangeably. Also, a cache will always be a TTL-based cache unless otherwise specified.

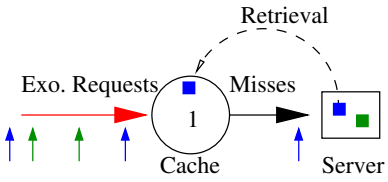


Figure 1: Single cache and server of two files *blue* and *green*.

We now introduce a key assumption for our approach:

Assumption 2.1 (Infinite Capacity)**.** *The TTL-based cache has an infinite capacity.*

A consequence is that content items are evicted from the cache only when their TTL expires and not because space is needed to allocate other contents. Assumption 2.1 allows us to decouple the management of the different content items and study each of them separately as illustrated in Figure 2. For this reason, in what follows we will refer to a *single* content item or data item. In Section 7.1, we show how a TTL-based cache can be implemented with constrained buffer and that our model can be used to compute good estimate of the performance metrics also in this more realistic case.

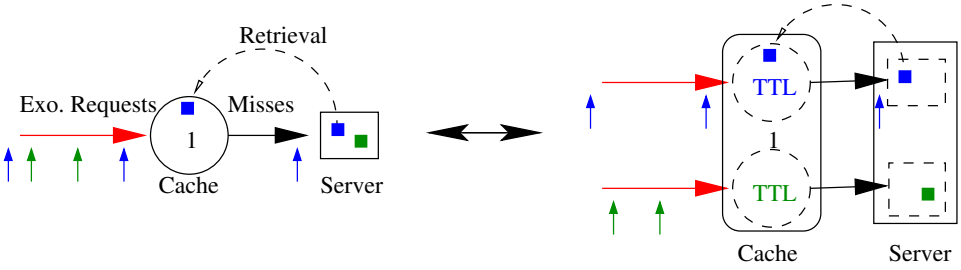


Figure 2: Infinite cache capacity and TTL decoupling effect.

In order to keep the model as simple as possible we also assume that data processing and transfer times are negligible:

Assumption 2.2 (Zero delay)**.** *There is a zero processing time at each node and a zero transmission delay between nodes including the server.*

We observe that the analysis does not change as long as the time to retrieve and process a data item is smaller than the inter-arrival time of two consecutive requests for the same item. The extension of the model to take into account *non-zero processing time* and/or *delay* will be investigated in future work.

Requests for a specific data item are generated at times $\{t_k, k \in \mathbb{Z}\}$ such that $\ldots < t_{-1} < t_0 \leq 0 < t_1 < \ldots$ by convention, where $\mathbb{Z}$ denotes the set of all integers. Let $X^{(k)} = t_{k+1} - t_k$ be the inter-arrival time between requests k and $k+1$. Also, let $T^{(k)}$ $(k \in Z)$ being the TTL duration generated for the content after the arrival of the request at time t_k.

Consider the request submitted at time t_0 (the process for requests submitted at times t_k with $k \neq 0$ is the same). There is a *cache hit* (resp. *cache miss*) at time t_0 if the data item is present (resp. is not present) in the cache at this time, which corresponds to the situation

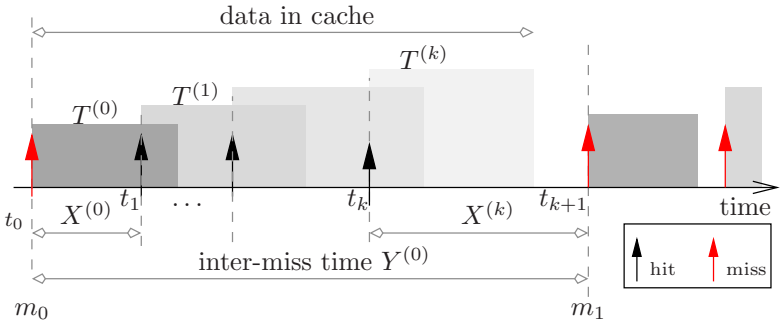


Figure 3: Requests, caching durations and inter-miss times.

where $t_0 \leq t_{-1} + T^{(-1)}$ (resp. $t_0 > t_{-1} + T^{(-1)}$). In the case of a cache miss the request is *instantaneously* (because of Assumption 2.2) forwarded to the server at time $m_0 = t_0$ and the data item is retrieved from the server. By convention, the data item is permanently stored in the server. Once the data item is fetched from the server, a copy of it is *instantaneously* transmitted to the cache and the request is resolved at time t_0, while a copy is kept at the cache. At time t_0 the TTL of the data item is set to $T^{(0)}$ both for a cache hit and for a cache miss. The next cache miss after time m_0 will occur at time $m_1 = t_j$ with $j = \min\{l > 0 : t_l > t_{l-1} + T^{(l-1)}\}$ – see Figure 3 where $j = l + 1$.

Our choice to reset the TTL after each cache hit makes also our single-cache scenario different from the one studied in [33]. Resetting the TTL tends to increase the sojourn time and the hit probability on the cache specially for the most popular contents. This corresponds to the general ICN paradigm [1] to move popular documents as close as possible to the users. For the time being we assume the following minimal assumptions:

Assumption 2.3 (Stationary arrivals and TTLs)**.** *The point process* $\{t_n, n \in \mathbb{Z}\}$ *is simple (i.e. there are no simultaneous requests) and stationary (i.e.* $\{X^{(k)}, k \in \mathbb{Z}\}$ *is a stationary sequence) and independent of the sequence of TTLs* $\{T^{(k)}, k \in \mathbb{Z}\}$ *which is also assumed to be stationary. Furthermore, the intensity* $\lambda := 1/\mathbb{E}[X^{(k)}]$ *of the point process* $\{t_n, n \in \mathbb{Z}\}$ *is non-zero and finite and* $0 < 1/\mu := \mathbb{E}[T^{(k)}] < \infty$.

Under Assumption 2.3 the cache is in steady-state (in particular) at time $t = 0$ and from now on we will only observe its behavior at times $t \geq 0$. We denote by $X(t) = \mathbb{P}(X^{(k)} < t)$ and $T(t) = \mathbb{P}(T^{(k)} < t)$ the Cumulative Distribution Function (CDF) of $X^{(k)}$ and $T^{(k)}$, respectively.

We call the *miss process* the sequence of successive time instants $0 \leq m_0 < m_1 < \cdots$ at which misses occur in $[0, \infty)$, which are also the times at which the server forwards a copy of the data item to the cache. We denote by $Y^{(k)} = m_{k+1} - m_k$ the time interval between the k-th and the $(k+1)$-st misses for $k \geq 0$ and $Y(t) = \mathbb{P}(Y^{(k)} < t)$ the CDF of $Y^{(k)}$. Stronger statistical assumptions on the sequences $\{X^{(k)}, k \in \mathbb{Z}\}$ and $\{T^{(k)}, k \in \mathbb{Z}\}$ will quickly become necessary only for the purpose of characterizing the miss process of the cache – see Assumption 3.1.

We will study the cache networks in their steady-state regime, and we will calculate the following performance metrics for each cache:

1. **the hit probability** H_P is the stationary probability that the content is in the cache when a new request arrives. The **miss probability** M_P is the complementary probability $1 - H_P$;
2. **the occupancy probability** O_P is the stationary probability that the content is in the cache at a random time instant;
3. **the miss rate** M_R is the rate at which the cache forwards requests to the server.

The hit probability is clearly a fundamental performance metric for a caching system. The occupancy probability is equal to the fraction of time that a content spends in the cache and then it can be used to characterize the instantaneous buffer occupancy distribution. Finally, for a single cache network the miss rate quantifies the load on the server, but for a hierarchical cache network a miss at one cache causes the request to be forwarded to higher-level caches. Hence, we need to characterize the miss process to be able to evaluate the hit probability at higher-level caches.

Before moving to the analysis of the single cache network, we need to introduce some more notation. For any non-negative random variable (rv) ξ with a CDF $F(t) = \mathbb{P}(\xi < t)$ $(\forall\ t \geq 0)$, we denote by

$$F^*(s) = \mathbb{E}[e^{-s\xi}] = \int_0^\infty e^{-st} F(\mathrm{d}t), \quad s \geq 0$$

the Laplace-Stieltjes Transform (LST) of ξ. The notation $F(\mathrm{d}t)$ is used because the Probability Density Function $f(t)$ of the rv ξ may not exist; otherwise, $F(\mathrm{d}t) = f(t)\mathrm{d}t$ as commonly seen. For any number $a \in [0, 1]$, $\bar{a} := 1 - a$ by definition. In particular, if $F(t)$ is a CDF, $\bar{F}(t) = 1 - F(t)$ is the corresponding Complementary Cumulative Distribution Function (CCDF).

From now on we assume that each cache satisfies Assumptions 2.1, 2.2 and 2.3.

λ	Arrival rate (single cache)
$1/\mu$	Expected TTL (single cache)
$X(t)$	CDF exogenous arrivals (single cache)
$Y(t)$	CDF inter-miss times (single cache)
$T(t)$	CDF TTL duration (single cache)
$Z^*(s)$	LST of CDF $Z(t)$
H_P, M_P	Hit, miss probability resp. (single cache)
H_R, M_R	Hit, miss rate resp. (single cache)
O_P	Occupancy probability (single cache)

Table 1: Notation for a single-cache network

3. Single TTL Cache: Analysis

Define $L(t) := \mathbb{P}(X^{(k)} < t, X^{(k)} < T^{(k)})$ the stationary probability that the inter-arrival time between two successive requests is smaller than t and smaller than the TTL associated with the former request. Because arrivals and TTLs are independent we have

$$L(t) = \int_0^t (1 - T(x))X(\mathrm{d}x),\ t \geq 0. \tag{1}$$

Using the notation in Table 1, the following proposition provides exact formulas for two of the performance metrics of interest.

Proposition 3.1 (Hit probability and miss rate)**.** *Under Assumption 2.3 the (stationary) hit probability H_P and the (stationary) miss rate M_R, are respectively given by*

$$H_P = \int_0^\infty (1 - T(x))X(\mathrm{d}x) = L(\infty), \tag{2}$$

$$M_R = \lambda(1 - H_P) \tag{3}$$

where we recall that $\lambda = 1/\mathbb{E}[X]$ is the request arrival rate.

Proof. The stationary hit probability H_P is defined as the probability that an arriving request finds the data item in the cache, i.e. the TTL has not expired yet, namely,

$$H_P = \mathbb{P}(X^{(k)} \leq T^{(k)}) = \int_0^\infty \mathbb{P}(x \leq T^{(k)})X(\mathrm{d}x) = \int_0^\infty (1 - T(x))X(\mathrm{d}x).$$

The stationary miss probability is $M_P = 1 - H_P$ so that the miss rate is given by (3). $\diamond$

Proposition 3.2 provides the *cache occupancy* (O_P) defined as the stationary probability that the data item is stored at the cache at a random time instant.

Proposition 3.2 (Occupancy probability)**.** *Under Assumption 2.3 the stationary cache occupancy O_P is given by*

$$O_P = \lambda \int_0^\infty (1 - T(t))(1 - X(t))\mathrm{d}t. \tag{4}$$

Proof. Let $\chi(t) \in \{0, 1\}$ be the cache occupancy at time t with $\chi(t) = 1$ if the data item is in the cache at time t and $\chi(t) = 0$ otherwise. Since under Assumption 2.3 the cache is in steady-state at time $t = 0$ we have $O_P = \mathbb{E}[\chi(0)] = \mathbb{P}(\chi(0) = 1)$.

Letting $Z(t) = \chi(t)$, $T_1 = X^{(0)}$ and $g(z) = \mathbf{1}(z > 0)$ in [2, Formula (1.3.2), p. 21] yields

$$O_P = \lambda \mathbb{E}^0 \left[\int_0^{X^{(0)}} \chi(t)\mathrm{d}t \right]$$

with $\mathbb{E}^0$ the expectation operator under the Palm probability $\mathbb{P}^0$ of the stationary point process $\{t_n, n \in \mathbb{Z}\}$ with associated marks $\{T^{(k)}, k \in \mathbb{Z}\}$. $\mathbb{P}^0$ has the property that $\mathbb{P}^0(t_0 = 0) = 1$ (see [2, Definition (1.2.1), p. 14]) which implies $\chi(t) = \mathbf{1}(t < T^{(0)})$ for $t \in [0, X^{(0)}]$ under $\mathbb{P}^0$. Hence,

$$\begin{aligned}
O_P &= \lambda \mathbb{E}^0 \left[\int_0^{X^{(0)}} \mathbf{1}(T^{(0)} > t)\mathrm{d}t \right] \\
&= \lambda \int_0^\infty \mathbb{E}^0 \left[\int_0^x \mathbf{1}(T^{(0)} > t)\mathrm{d}t \right] X(\mathrm{d}x) \qquad (5) \\
&= \lambda \int_0^\infty \left(\int_0^x (1 - T(t))\mathrm{d}t \right) X(\mathrm{d}x) \\
&= \lambda \int_0^\infty (1 - T(t)) \left(\int_t^\infty X(\mathrm{d}x) \right) \mathrm{d}t \\
&= \lambda \int_0^\infty (1 - T(t))(1 - X(t))\mathrm{d}t
\end{aligned}$$

where (5) is obtained by conditioning on the rv $X^{(0)}$ with CDF $X(t)$ and by using the independence of the rvs $X^{(0)}$ and $T^{(0)}$. This completes the proof. $\diamond$

Notice that the hit probability H_P and the occupancy probability O_P differ in general. They are equal if the arrival process $\{t_n, n \in \mathbb{Z}\}$ is a Poisson process thanks to the *PASTA* property.

To state the next results we need to strengthen the statistical assumptions made on the sequences $\{X^{(k)}, k \in \mathbb{Z}\}$ and $\{T^{(k)}, k \in \mathbb{Z}\}$.

Assumption 3.1 (Renewal arrivals and TTLs)**.** *Both sequences* $\{X^{(k)}, k \in \mathbb{Z}\}$ *and* $\{T^{(k)}, k \in \mathbb{Z}\}$ *are mutually independent renewal sequences.*

Assumption 3.1 is general enough to cover a broad range of applications. In his earlier work, Whitt [47] developed two basic methods to approximate a point process with a renewal process, and he showed in a joint work with Feldmann [17] that the long-tailed distributions which are generally observed in network performance analysis can be fitted by a renewal process with a hyper-exponential inter-arrival distribution. Also, Jung et *al.* [32] used renewal processes with Weibull and Pareto CDFs to fit the traces of DNS servers at the MIT Computer Science and Artificial Intelligence Laboratory and the Korea Advanced Institute of Science and Technology. More recently, Nelson and Gerhardt [41] have surveyed different methods used to fit a general point process to a special Phase-Type renewal process via *Moment Matching* techniques. In contrast to many existing works where it is assumed that the arrival process obeys to the Independent Reference Model (IRM) (or equivalently [20] that the arrival process is a Poisson process), our renewal assumption 3.1 is less restrictive.

We now evaluate the CDF $Y(t)$ of the miss process which will be needed to extend the analysis to a network of caches since a cache may receive requests due to misses at lower-level caches.

Proposition 3.3 (CDF of the miss process)**.** *Under Assumption 3.1 the miss process of a single cache is a renewal process. The CDF* $Y(t)$ *of the inter-miss times is the solution of the integral equation*

$$Y(t) = X(t) - L(t) + \int_0^t Y(t-x)\mathrm{d}L(x) \tag{6}$$

or, in compact form, $Y = X - L + L \star Y$ *with* $\star$ *denoting the convolution operator. The renewal equation (6) has one and only one bounded solution given by* $Y = R \star (X - L)$ *where* $R = \sum_{n \geq 0} L^{(n)}$ *and* $L^{(n)}$ *denotes the nth-fold convolution of the function* L *with itself (by convention* $L^{(0)} \equiv 1$*).*

The LST $Y^*(s)$ *of the inter-miss times is given by*

$$Y^*(s) = \frac{X^*(s) - L^*(s)}{1 - L^*(s)}. \tag{7}$$

Proof.

Without loss of generality, assume that the first request arrives at time $t_0 = 0$ and finds an empty cache. Since a miss triggers a new TTL, miss times are regeneration points for the state of the cache under Assumption 3.1. This implies that miss instants form a renewal process which is fully characterized by the CDF $Y(t)$ of the generic inter-miss time denoted by Y.

The rest of the proof is an adaptation of a classical argument in renewal theory (see [12, Chapter 9]). Recall that $X^{(0)}$ (resp. $Y^{(0)}$, $T^{(0)}$) denotes the first inter-arrival time (resp. inter-miss time, TTL value) after $t_0 = 0$ as shown in Fig. 3. Since $Y^{(0)} \geq X^{(0)}$ the event $\{Y^{(0)} < t\}$ may only occur if $X^{(0)} < t$. Therefore,

$$\begin{aligned} Y(t) &= \mathbb{P}(Y^{(0)} < t, T^{(0)} < X^{(0)} < t) + \mathbb{P}(Y^{(0)} < t, T^{(0)} > X^{(0)}, X^{(0)} < t) \\ &= \mathbb{P}(T^{(0)} < X^{(0)} < t) + \mathbb{E}\left[Y(t - X^{(0)})\mathbf{1}(T^{(0)} > X^{(0)}, X^{(0)} < t)\right] \\ &= \mathbb{P}(X^{(0)} < t) - \mathbb{P}(X^{(0)} < t, X^{(0)} < T^{(0)}) + \int_0^t Y(t-x)\mathbb{P}(T^{(0)} > x)X(\mathrm{d}x) \qquad (8) \end{aligned}$$

where we have used the independence of $X^{(0)}$ and $T^{(0)}$ to establish (8). Then it follows from equation (1) that

$$\begin{aligned} Y(t) &= X(t) - L(t) + \int_0^t Y(t-x)(1 - T(x))X(\mathrm{d}x) \\ &= X(t) - L(t) + \int_0^t Y(t-x)\mathrm{d}L(x) \qquad (9) \end{aligned}$$

The renewal equation (9) may be written $Y = X - L + L \star Y$. It is well known that its solution exists and is unique and is given by $Y = R \star (X - L)$ where $R = \sum_{n \geq 0} L^{(n)}$ [12, Theorem 2.3, p. 294].

From the identity $Y = X - L + L \star Y$ we readily get (7), which concludes the proof. ◇

Approximation for LRU caches. Che et *al.* [10] have experimentally shown that LRU caches fed with requests described by Poisson processes can be accurately modeled as deterministic TTL-based cache in isolation. In a 2013 paper, Martina and co-authors [11] have extended these experimental results [10] to the case of renewal request processes. The constant TTL value D is referred to in [10, 11] as the *characteristic time* of the LRU cache, and it is obtained by solving a

fixed-point equation. A similar fixed-point equation is derived when Assumption 2.1 is removed i.e. in the case of finite cache capacity as shown in Section 7.2. Given a deterministic TTL value D, we have $T(t) = \mathbf{1}(t > D)$ and Equations (2), (3), (4) and (6) become

Corollary 3.1 (Deterministic TLLs)**.**

$$H_P = X(D) \ , \ M_R = \lambda(1 - X(D)) \ , \ O_P = \lambda \int_0^D (1 - X(x))\mathrm{d}x \tag{10}$$

$$Y(t) = \mathbf{1}(t > D) \left(X(t) - X(D) + \int_0^D Y(t - x)X(\mathrm{d}x) \right) \tag{11}$$

Remark 3.1 (Counters of Particles)**.** *A single cache with a deterministic TTL is called a Geiger counter of type II in [12, Example (1.34), p. 292].*

Approximation for RND caches. It was experimentally shown in [11] that RND caches can be studied as *memoryless* TTL-based caches with exponentially distributed TTLs. Also in this case, the expected TTL value μ^{-1} is solution of a fixed point equation as derived in Section 7.1. In this case $T(t) = 1 - e^{-\mu t}$, $L^*(s) = X^*(s + \mu)$, the metrics of interest and the miss process characterization follow directly from Equations (2), (3), (4) and (7).

Corollary 3.2 (Exponential TLLs)**.**

$$H_P = X^*(\mu) \ , \ M_R = \lambda(1 - X^*(\mu)) \ , \ O_P = \frac{\lambda(1 - X^*(\mu))}{\mu} \tag{12}$$

$$Y^*(s) = \frac{X^*(s) - X^*(s + \mu)}{1 - X^*(s + \mu)}. \tag{13}$$

Applying standard results about convex ordering and the formulas above, we obtain the following interesting property for a deterministic TTL cache, that is proven in the Appendix.

Proposition 3.4. *Given the expected TTL value $D = \mathbb{E}[T]$ and the CDF $X(t)$ of inter-arrival times, the occupancy O_P is maximized when the TTL is deterministic and equal to D. Moreover, if $X(t)$ is a concave function then the hit probability H_P is maximized too.*

Proposition 3.4 theoretically explains the superiority, in terms of hit and occupancy probabilities, of LRU caches over RND caches (given that $D = \mu^{-1}$) when they are fed by IRM traffic (or Poisson processes [20]) or traces in [17, 32, 33] fitted by renewal processes. We can easily check that the CDFs $F(t)$ of inter-arrival times in these experiments are concave functions.

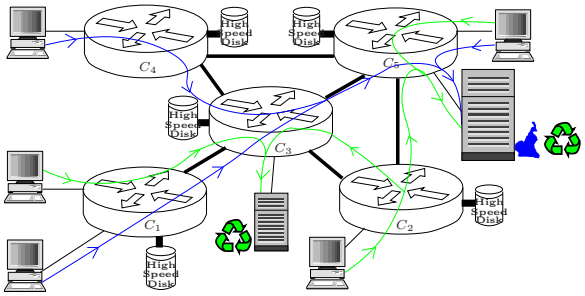


Figure 4: Requests for the *green* file are routed on as polytree, while those for the *blue* file are routed as tree.

4. Hierarchical TTL-based Cache networks

In this section, we consider TTL-based cache networks fed by exogenous renewal request processes. We maintain Assumptions 2.1 and 2.2, i.e. we assume each cache has an infinite buffer and processing and transmission delays in the network are negligible. Hence, files at each cache are decoupled and can be separately studied. Therefore, we focus on a single file whose requests are propagated along a tree, but it is easy to generalize to a polytree. We draw the attention of the reader to the fact the cache network topology has not to be a tree (or a polytree); in fact, it can be general because files are decoupled and requests can be independently routed (see the example in Figure 4).

We consider then a tree of caches: the root is connected to the content server, each other cache has a parent cache to which it forwards all the requests which cannot be satisfied locally. Once located, the data item is routed on the reverse-path and a copy is placed in each cache. Each cache operates exactly as described in the previous section, by setting a TTL for each new data item stored and redrawing the TTL at each cache hit. We denote by $\mathcal{C}(n)$ the set of children of cache n. The following assumption holds:

Assumption 4.1 (TTL Values)**.** *TTL values extracted at each cache are i.i.d. values. We denote by $T_n(t)$ the CDF of TTL values at cache n. TTL values at different caches are independent and they are also independent from the request arrival processes.*

The ICN content-routers [1] are examples of caches that behave independently of other caches and of the requests they receive. They also decide locally what content to store or discard at least as described in the paper of Van Jacobson et *al.* [28].

λ_n	Exogenous arrival rate at cache n
Λ_n	Total arrival rate at cache n
$1/\mu_n$	Expected TTL at cache n
$X_n(t)$	CDF exogenous arrivals at cache n
$Z_n(t)$	CDF overall arrivals at cache n
$Y_n(t)$	CDF inter-miss times at cache n
$T_n(t)$	CDF TTL duration at cache n
$H_{P,n}, M_{P,n}$	Hit, miss probability resp. at cache n
$H_{R,n}, M_{R,n}$	Hit, miss rate resp. at cache n
$O_{P,n}$	Occupancy of cache n (stationary probability content is in cache n
$\mathcal{C}(n)$	Set of children of cache n
$X^*(s)$	LST of CDF $X(t)$

Table 2: Glossary of main notation for cache n

Each cache, say cache n, receives two flows of requests: users' requests arriving directly at a cache are called *exogenous* and form the *exogenous* arrival process, misses at the children caches in $\mathcal{C}(n)$ form the *endogenous* arrival process. We generalize Assumption 3.1 as follows:

Assumption 4.2 (Exogeneous Request arrivals are Renewal Processes)**.** *The exogeneous arrival processes are independent renewal processes. We denote by $X_n(t)$ the CDF of the inter-arrival times of exogenous requests at cache n.*

Similarly, we add the subscript n to the quantities defined in Section 3 to denote the same quantities at cache n ($Y_n(t)$, $L_n(t)$, λ_n, $H_{P,n}$, $M_{R,n}$ and $O_{P,n}$). The superposition of the exogenous and endogenous arrival processes at cache n forms the *aggregated* arrival process. We introduce $Z_n(t)$ and Λ_n to denote respectively its inter-arrival time CDF and its rate. The notation is summarized in Table 2.

The exact analysis we carried on in the previous section can be extended immediately to the case of *linear-star networks*, that we study in the section below.

4.1. Exact analysis of Linear-star networks

A linear cache network is a tandem of caches and one server, where exogeneous requests arrive only to the cache farthest from the server as illustrated in Figure 5. Since the arrival process at

the first cache is a renewal process (Assumption 3.1) the miss process of the first cache is also a renewal process (Proposition 3.3). Therefore, the second cache is fed by a renewal process. Reasoning in this way iteratively, we can then show that all the caches are fed by a renewal process. Moreover, all the metrics of interest can be derived successively at each cache from the results on a single cache analysis in Section 3.

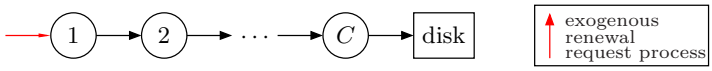


Figure 5: Linear cache network.

A star cache network is a tree with one internal node i.e. the root and leaves (see Figure 6). For this one-level tree, the metrics of interest and the miss process are easily found at each leaf from the single cache analysis in Section 3. The miss processes are also renewal processes since the request processes at the leaves are renewal processes. Hence, the root is fed by the aggregated request process resulting from the superposition of the (renewal) miss processes and its exogenous renewal process. It is possible to calculate exactly the CDF of the first inter-arrival time in the aggregated arrival process (see Theorem 4.1 and following remarks), then the metrics of interest are obtained from Propositions 3.1 and 3.2 since the aggregated process is a stationary process. Our analysis provides exact results on star networks of caches.

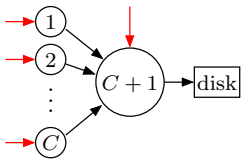


Figure 6: Star cache network.

We generalize our exact approach on these two networks topologies by defining a class of networks called *Linear-star cache network* illustrated in Figure 7. We can characterize the exact performance metrics on any network that belongs to this class as follows: we start from the leaves and apply our Propositions 3.1, 3.2 and 3.3 (as was done for the linear network), until we reach the root where we apply Theorem 4.1 and Propositions 3.1 and 3.2 (as was described for the star network).

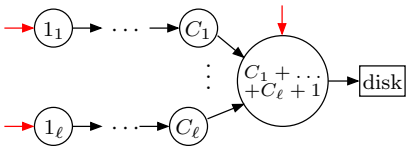


Figure 7: Linear-star cache network.

4.2. Approximated methodology for general tree networks

The approach we described in Section 4.1 cannot be extended to arbitrary hierarchical networks. The problem arises from the fact that the aggregated arrival process is not in general a renewal process (this is not the case even if the exogenous and endogenous arrival processes are both renewal ones). Hence, we cannot apply Proposition 3.3 that allows us to characterize the exact miss process. Indeed in the linear-star cache network we cannot extend our analysis beyond the cache with more than one child. Nevertheless, for our analysis we will suppose that the aggregated request process is a renewal process and we make the following approximation:

Approximation 4.1 (Overall Process)**.** *The overall (aggregated) arrival process at node n is approximated by a renewal process with inter-arrival time CDF $Z_n(t)$.*

Note that the statement above has a different status than the other assumptions in this paper. While the assumptions can be considered approximations for actual cache networks, they are internally consistent. On the contrary the statement in Approximation 4.1 is in general false, even in the framework of our model. Nevertheless, it makes the analysis possible and leads to excellent approximations as we are going to show later.

We have shown in Section 3 that the miss process of a TTL cache fed by a renewal process is itself a renewal process, then a corollary of Approximation 4.1 is that each miss process can be considered a renewal process. In this case, the aggregated arrival process is the superposition of independent (due to the tree topology of the network) renewal processes and the CDF $Z_n(t)$ of inter-arrival times has been calculated by Lawrence [40, Formula (4.1)].

Theorem 4.1. *The CDF $A(t)$ of the first inter-event time of the point process resulting from the superposition of K independent renewal processes is given by*

$$A(t) = 1 - \sum_{k=1}^{K} \frac{\alpha_k}{\sum_{l=1}^{K} \alpha_l} \left(1 - A_k(t)\right) \prod_{j=1, j\neq k}^{K} \alpha_j \int_t^{\infty} \left(1 - A_j(u)\right) \mathrm{d}u,$$

where $A_k(t)$ and $\alpha_k > 0$ are respectively the inter-event time CDF and the arrival rate of the k^{th} process.

Theorem 4.1 actually holds for the superposition of independent stationary point processes [2, Formula (1.4.6), p. 35]. Note in passing that such a superposition is itself a stationary process, which allows us to use formulas (2), (3) and (4) to compute the hit probability, miss rate and cache occupancy, respectively, of a cache fed by the superposition of independent stationary processes, and then in particular of renewal ones.

Thanks to Approximation 4.1 and Theorem 4.1, we are ready to study any cache tree network. The total request rate Λ_n at a node n, is

$$\Lambda_n = \lambda_n + \sum_{i \in \mathcal{C}(n)} M_{R,i} \tag{14}$$

where $\mathcal{C}(n)$ is the set of children of node n. Since the exogenous process (with CDF $X_n(t)$) and the miss process at the i-th child of node n (with CDF $Y_i(t)$) are renewal processes, we invoke Theorem 4.1 to compute the approximate inter-arrival times CDF $Z_n(t)$ of the overall arrival process. We get

$$\begin{aligned} Z_n(t) &= 1 - \frac{\lambda_n}{\Lambda_n} \bar{X}_n(t) \prod_{i \in \mathcal{C}(n)} M_{R,i} \int_t^\infty \bar{Y}_i(u) \mathrm{d}u \\ &\quad - \sum_{i \in \mathcal{C}(n)} \frac{M_{R,i}}{\Lambda_n} \bar{Y}_i(t) \lambda_n \int_t^\infty \bar{X}_n(u) \mathrm{d}u \times \prod_{\substack{j \in \mathcal{C}(n) \\ j \neq i}} M_{R,j} \int_t^\infty \bar{Y}_j(u) \mathrm{d}u. \end{aligned} \tag{15}$$

The approximate inter-miss times CDF $Y_n(t)$ at cache n is obtained from Proposition 3.3 since we approximate the overall request process by a renewal process with CDF $Z_n(t)$ by Approximation 4.1.

$$Y_n(t) = Z_n(t) - L_n(t) + \int_0^t Y_n(t-x) \mathrm{d}L_n(x) \tag{16}$$

where $L_n(t) = \int_0^t (1 - T_n(x)) \mathrm{d}Z_n(x)$ and $T_n(t)$ is the CDF of the TTL duration at cache n.

Equations (14), (15) and (16) provide a recursive procedure for calculating, at least numerically, the request rate Λ_n and the approximate CDFs $Y_n(t)$ and $Z_n(t)$ at each cache n of an arbitrary hierarchical network starting from the leaves. The approximate metrics of interest are

obtained from Propositions 3.1 and 3.2. Our approach is summarized in the following algorithm:

Algorithm 1: General Procedure on tree fed by general renewal processes

input : `TreeDepth` d, `CDFs` $X_n(t)$, $\{Y_i(t), i \in \mathcal{C}(n)\}$ `and` $\{T_n(t), n \geq 1\}$

output: `Metrics of interest` Λ_n, $H_{P,n}$, $O_{P,n}$ `and CDF` $Y_n(t)$

while $d \neq 0$ **do** ; `// Caches are different from the server`

  foreach n `in the set of caches at depth` d **do** ; `// Start from Leaves`

    $\Lambda_n \xleftarrow{\text{Eq(14)}} \{\lambda_n, M_{R,i}, i \in \mathcal{C}(n)\}$;

    if $\mathcal{C}(n) = \emptyset$ **then**

      $Z_n(t) \leftarrow X_n(t)$;

    else

      $Z_n(t) \xleftarrow{\text{Eq(15)}} \{X_n(t),\ Y_i(t), i \in \mathcal{C}(n)\}$;

    end

    $H_{P,n},\ M_{R,n} \xleftarrow{\text{Prop.(3.1)}} \{Z_n(t),\ T_n(t)\}$;

    $O_{P,n} \xleftarrow{\text{Prop.(3.2)}} \{Z_n(t),\ T_n(t)\}$;

    $Y_n(t) \xleftarrow{\text{Eq(16)}} \{Z_n(t),\ T_n(t)\}$;

  end

  $d \leftarrow d - 1$;

end

While this algorithm allows us to study any cache tree under any possible exogenous arrival processes and TTL distributions, its numerical complexity can be very high as it requires to evaluate some integrals over infinite ranges as in (15) and to solve an integral equation as in (16). As we are going to show, simpler algorithms exist for more specific distributions.

4.3. Hierarchical networks with Matrix-Exponential request inter-arrival times and TTL

We consider a hierarchical cache network where the inter-arrival times of exogenous requests and the TTL values are described by Matrix-Exponential (ME) distributions, i.e. whose CDFs and

PDFs are defined by

$$\Psi(t) = 1 - \boldsymbol{\alpha}\ e^{St}\ \mathbf{1}_R\ ,\ \psi(t) = \boldsymbol{\alpha}\ e^{St}\ (-S\mathbf{1}) \quad t \geq 0 \tag{17}$$

respectively [26], where $\boldsymbol{\alpha}$ is a 1-by-R vector, called the starting vector, S is an R-by-R matrix, called the progress rate matrix, and $\mathbf{1}_R$ is an R-by-1 vector whose elements are all equal to 1. In general different pairs $(\boldsymbol{\alpha}, S)$ can lead to the same CDF $\Psi(t)$. Here, we consider a representation with minimal order R. In [26, Theorem 3.1], He and Zhang established under which conditions R is minimal and they showed that in this case the matrix S is a Jordan matrix and Ψ can be written as follows (Karlin's representation [34]):

$$\Psi(t) \quad = \quad 1 - \sum_{k=1}^{K} Q_k(t) e^{\sigma_k t},\ t \geq 0 \tag{18}$$

where $\{\sigma_k\}_k$ are the eigenvalues of S, $\sigma_i \neq \sigma_j$ if $i \neq j$, $Q_k(t) = \sum_{j=0}^{r_k - 1} q_{k,j} t^j$ is a polynomial of degree $r_k - 1$ and $\sum_{k=1}^{K} r_k = R$. The relations between $\boldsymbol{\alpha}$ and $\{q_{k,j}, 1 \leq k \leq K, 0 \leq j \leq r_k - 1\}$ can be found in [26]. In what follows we will usually consider Karlin's representation (18). The class of ME distributions is equivalent to the class of distributions having a rational LST [26], it includes then also all the phase-type distributions (i.e. any mixture of exponential distributions). In what follows we are going to call a request renewal process with ME distributed inter-arrival times simply an ME renewal process. Similarly, we are going to use the expression ME TTL to indicate TTLs that are ME distributed.

The following result guarantees us that if the request arrival process at a cache is an ME renewal process and TTL are ME distributed, then the miss process is also a ME renewal process with a known representation.

Proposition 4.1 (ME miss process)**.** *If the TTLs and the inter-arrival times of the request renewal process at cache n are ME distributed, then the miss inter-arrival times are ME distributed.*

Proof. We consider a cache n where the inter-arrival times and the TTLs are characterized by the ME CDFs $X_n(t)$ and $T_n(t)$. Both $X_n(t)$ and $T_n(t)$ admit a Karlin representation and a rational LST. From the definition (1) also $L(t)$ has a Karlin representation and its LST is

rational. Thus, the solution $Y_n(t)$ in (6) is a CDF with a rational LST $Y_n^*(s)$ given by

$$Y_n^*(s) \quad = \quad 1 - \frac{1 - X_n^*(s)}{1 - L_n^*(s)} = 1 - \frac{N(s)}{D(s)}$$

where $N(s)$ and $D(s)$ are the numerator and the denominator of the fraction $1 - Y_n^*(s)$ after factorization and simplification of common terms. The CDF $Y_n(t)$ is a ME distribution by the equivalence between ME distributions and CDFs having rational LST. Moreover, $Y_n(t)$ admits a Karlin representation:

$$Y_n(t) = 1 - \sum_{k=1}^{K} \sum_{j=0}^{r_k - 1} q_{k,j} t^j e^{\sigma_k t}, \ t > 0 \tag{19}$$

where the exponents $\{\sigma_k\}_k$ are the zeros of $D(s)$ and the coefficients $\{r_k, q_{k,j}\}_{k,j}$ are given by the relations

$$q_{k,r_k - i} \quad = \quad \frac{1}{i!} \left\{ \frac{d^i}{ds^i} \left[(1 - Y^*(s))(s - \sigma_k)^{r_k} \right] \right\}_{s=\sigma_k}. \tag{20}$$

$\diamond$

In general the aggregated request arrival process at cache n is the superposition of the miss processes at the cache in $\mathcal{C}(n)$ and the exogenous arrival process. If each of these processes is a ME renewal process, Approximation 4.1 and Theorem 4.1 allows us to conclude that also the inter-arrival times of the aggregated request arrival process are ME distributed. Under the Approximation 4.1 and Proposition 4.1, all the miss processes in the network are ME renewal processes. We can characterize them iteratively starting from the leaves as for the general case.

4.4. Hierarchical networks with Diagonal Matrix-Exponential request inter-arrival times and TTLs

The calculations become even simpler when the progress rates matrices (S) of the ME distributions are diagonal or diagonalizable. In this case, the distribution is said to be *Diagonal Matrix-Exponential*. Without loss of generality, if $\Psi(t)$ is a Diagonal ME distribution, then its Karlin representation is:

$$\Psi(t) = 1 - \sum_{j=1}^{K} q_j e^{\sigma_j t}.$$

If the request arrival process at cache n is a Diagonal ME renewal process and the TTLs are

Diagonal ME distributed, respectively with CDFs

$$X_n(t) = 1 - \sum_{k=1}^{K_n} a_{n,k} e^{-\sigma_{n,k} t} \quad \texttt{and} \quad T_n(t) = 1 - \sum_{j=1}^{J_n} b_{n,j} e^{-\mu_{n,j} t},$$

then the metrics of interests at cache n are obtained from a straightforward calculation by applying Propositions 3.1 and 3.2.

Corollary 4.1 (Metrics of interests at a Diagonal ME cache)**.** *The request rate λ_n, the hit probability $H_{P,n}$, the miss rate $M_{R,n}$ and the occupancy probability $O_{P,n}$ at cache n are calculated with the formulas*

$$\lambda_n = \sum_{k=1}^{K_n} a_{n,k}\sigma_{n,k}, \qquad H_{P,n} = \textstyle\sum_{j=1}^{J_n} b_{n,j} X_n^*(\mu_{n,j}) \tag{21}$$

$$M_{R,n} = \lambda_n \left(1 - \sum_{j=1}^{J_n} b_{n,j} X_n^*(\mu_{n,j})\right), \quad O_{P,n} = \lambda_n \textstyle\sum_{j=1}^{J_n} b_{n,j} \left(\frac{1 - X_n^*(\mu_{n,j})}{\mu_{n,j}}\right). \tag{22}$$

Similarly, the miss process is characterized by applying Propositions 3.3 and 4.1.

Corollary 4.2 (Miss process at a Diagonal ME cache)**.** *The LST of inter-miss times CDF is*

$$Y_n^*(s) = 1 - (1 - X_n^*(s)) \left(1 - \sum_{j=1}^{J_n} b_{n,j} X_n^*(s + \mu_{n,j})\right)^{-1} \tag{23}$$

which can be inverted as

$$Y_n(t) = 1 - \left[\sum_{k=1}^{K_n} a_{n,k} \left(1 + \sum_{i=1}^{K_n \times J_n} \frac{\gamma_i}{\theta_i - \sigma_{n,k}}\right) e^{-\sigma_{n,k} t} + \sum_{i=1}^{K_n \times J_n} \frac{-\gamma_i}{\theta_i} \left(1 + \sum_{k=1}^{K_n} \frac{a_{n,k}\sigma_{n,k}}{\theta_i - \sigma_{n,k}}\right) e^{-\theta_i t}\right] \tag{24}$$

where $(\theta_i)_{1 \le i \le K_n \times J_n}$ are solutions of the algebraic equation in z

$$0 = 1 - \sum_{i=1}^{K_n \times J_n} \frac{\delta_i}{\eta_i - z}, \tag{25}$$

$(\gamma_i)_{1\leq i\leq K_n\times J_n}$ *is the vector solution of the linear system of algebraic equations:*

$$\left\{0 = 1 + \sum_{i=1}^{K_n\times J_n} \frac{\gamma_i}{\theta_i - \eta_l}\right\}_{l=1}^{K_n\times J_n}, \tag{26}$$

the constants δ_i *and* η_i *are given by*

$$\delta_i = a_{n,k}b_{n,j}\sigma_{n,k} \quad , \quad \eta_i = \sigma_{n,k} + \mu_{n,j} \tag{27}$$

and (k, j) *is the* i^{th} *couple according to some ordering of the product set* $\{1, \ldots, K_n\}\times\{1, \ldots, J_n\}$. *Note also that the CDF* $Y_n(t)$ *is a Diagonal ME distribution.*

When we superpose Diagonal ME renewal processes, the inter-arrival times of the superposed process are still Diagonal ME distributed. In particular if Y_i is the Diagonal ME miss process at cache $i \in \mathcal{C}(n)$, with CDF

$$Y_i(t) = 1 - \sum_{k=1}^{K_i} a_{i,k}e^{-\sigma_{i,k}t}.$$

the overall arrival process is then characterized in the following corollary of Proposition 4.1:

Corollary 4.3 (Overall Request Process at Diagonal ME cache)**.** *Under Assumptions 4.1 and 4.1, the CDF of inter-arrival times* $Z_n(t)$ *in the overall request process at cache* n *is given by*

$$Z_n(t) = 1 - \sum_{i\in\mathcal{C}(n)\cup\{n\}} \sum_{k=1}^{K_i} \frac{a_{i,k}}{\Lambda_n} \left(\lambda_n \times \prod_{j\in\mathcal{C}(n)} M_{R,j}\right) e^{-\sigma_{i,k}t} \prod_{\substack{j\in\mathcal{C}(n)\cup\{n\}\\ j\neq i}} \left(\sum_{k=1}^{K_j} \frac{a_{j,k}}{\sigma_{j,k}} e^{-\sigma_{j,k}t}\right) \tag{28}$$

where $M_{R,i}$ *is the miss rate of the* i^{th} *child node,* λ_n *is the rate of exogenous requests, and* Λ_n *is the total request rate at the cache* n.

The Algorithm 1 simplifies when all the TTLs and the exogeneous request arrival processes are Diagonal ME and becomes:

Algorithm 2: Efficient Procedure on Diagonal MED cache tree fed by Diagonal ME renewal processes

input : `TreeDepth` d, `CDFs` $X_n(t)$, $\{Y_i(t), i \in \mathcal{C}(n)\}$ `and` $\{T_n(t), n \geq 1\}$

output: `Metrics of interest` Λ_n, $H_{P,n}$, $O_{P,n}$ `and CDF` $Y_n(t)$

while $d \neq 0$ **do** ; `// Caches are different from the server`

  foreach n `in the set of caches at depth` d **do** ; `// Start from Leaves`

    $\Lambda_n \xleftarrow{\text{Eq.(14)}} \{\lambda_n, M_{R,i}, i \in \mathcal{C}(n)\}$;

    if $\mathcal{C}(n) = \emptyset$ **then**

      $Z_n(t) \leftarrow X_n(t)$;

    else

      $Z_n(t) \xleftarrow{\text{Eq.(28)}} \{X_n(t),\ Y_i(t), i \in \mathcal{C}(n)\}$;

    end

    $H_{P,n},\ M_{R,n},\ O_{P,n} \xleftarrow{\text{Cor.(4.1)}} \{Z_n(t),\ T_n(t)\}$;

    $Y_n(t) \xleftarrow{\text{Cor.(4.2)}} \{Z_n(t),\ T_n(t)\}$;

  end

  $d \leftarrow d - 1$;

end

Before concluding our theoretical analysis and moving to the validation of our approximations, we observe that it is possible to adapt the formulas above for Diagonal ME exogeneous processes and TTLs to consider a slightly larger class of networks—initially introduced in [7] and denoted class $\mathcal{N}$ networks—that extends hierarchical Diagonal ME cache networks as follows: a single exogeneous request process is allowed to be a renewal process with a general CDF (not necessarily a Diagonal ME distribution) for inter-arrival times. We do not develop the general procedure for class $\mathcal{N}$, but we show how the formulas change for a specific case. Consider that the endogeneous request process at cache n is a general renewal process with CDF $Y_i(t)$ and rate $M_{R,i}$ (we use this notation as the misses were all generated at a child cache i), while the exogeneous request process at cache n has Diagonal ME CDF $X_n(t) = 1 - \mathbf{a}_n e^{-A_n t}\mathbf{1}_{K_n}$ and arrival rate $\lambda_n = (\mathbf{a}_n(-A_n)^{-1}\mathbf{1}_{K_n})^{-1}$. The total request rate at node n is $\Lambda_n = M_{R,i} + \lambda_n$. The following proposition generalizes the CDF of inter-arrival times of the aggregated request process $Z_n(t)$.

Proposition 4.2 (Approximation for class $\mathcal{N}$). *The performance metrics at cache n of a class $\mathcal{N}$ network are obtained from Corollary 4.1, and the overall request process is characterized by the CDF $Z_n(t)$ in (15) whose LST $Z_n^*(s)$ is given by*

$$Z_n^*(s) = 1 - s\frac{\lambda_n}{\Lambda_n}\sum_{k=1}^{K_n}\frac{a_{n,k}}{s+\sigma_{n,k}} - s^2\frac{\lambda_n M_{R,i}}{\Lambda_n}\sum_{k=1}^{K_n}\frac{a_{n,k}(1-Y_i^*(s+\sigma_{n,k}))}{(s+\sigma_{n,k})^2\sigma_{n,k}} \tag{29}$$

Proof. Applying the (15) to the CDFs $Y_i(t)$ and $X_n(t) = 1 - \sum_{k=1}^{K_n} a_{n,k}e^{-\sigma_{n,k}t}$, we obtain

$$Z_n(t) \;=\; 1 - \frac{\lambda_n M_{R,i}}{\Lambda_n}\sum_{k=1}^{K_n} a_{n,k} \times \left(\bar{Y}_i(t)\int_t^\infty e^{-\sigma_{n,k}u}\mathrm{d}u + e^{-\sigma_{n,k}t}\int_t^\infty \bar{Y}_i(u)\mathrm{d}u\right).$$

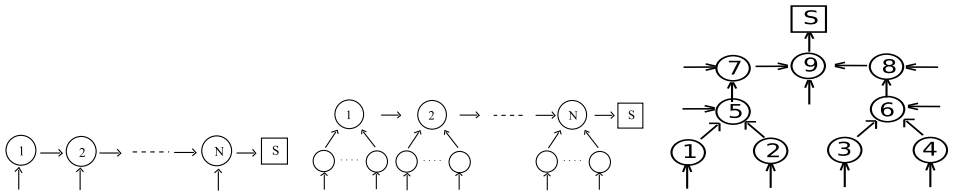
We deduce (29) by taking the LST of the latter equation. The metrics of interest and the LST $Y_n^*(s)$ are obtained by replacing the LST $X_n^*(s)$ by $Z_n^*(s)$ in Corollary 4.1 and Corollary 4.2. $\diamond$

5. Validation and Numerical Results

In this section, we investigate the accuracy of Approximation 4.1 and then of the approximate results obtained through Algorithms 1 and 2. We recall that Approximation 4.1 consists in considering that all aggregated request processes are renewal processes.

We evaluate the approximation quality by simulations in tree networks. In particular we focus first on networks of exponentially distributed TTL-based caches fed by requests generated according to Poisson processes for which it is possible to carry on an exact analysis (Section 5.1); then we look at a tree of deterministic TTL-based caches also fed by Poisson requests (Section 5.2); and finally we investigate the situation where requests are described by more general renewal processes and TTL distributions (section 5.3).

Before presenting these results, we mention that for a tandem of two caches, there exists an exact expression for the first autocorrelation lag ($\texttt{ACF}_1$) of the aggregated process at node 2 [40, Eq.(6.4)]. If the aggregate process were a renewal one, this lag would be identically zero. In reality it is non-null and depends on the arrival rates λ_1 and λ_2 and the timer μ_1. We find that for any possible choice of these parameters $0 > \texttt{ACF}_1 > -0.015$. Simulation results show that the autocorrelation is even less significant at larger lags. Therefore, inter-arrival times are weakly coupled and Approximation 4.1 is indeed accurate in this small network scenario.



(a) Linear network (b) Caterpillar network (c) Balanced tree network

Figure 8: Simulated Networks

5.1. Poisson traffic and Exponential timers

We start by observing that when the exogenous request processes are Poisson processes, it is possible to model a tree network of N caches as an irreducible continuous time Markov process, with state $\mathbf{x}(t) = (x_1(t), \ldots, x_N(t)) \in \mathcal{E} = \{0,1\}^N$, where $x_n(t) = 1$ (resp. $x_n(t) = 0$) if the data item is present (resp. missing) at time t at node n. Once the steady-state probabilities $(\pi(\mathbf{x}),\ \mathbf{x} \in \mathcal{E})$ have been calculated, exact values of the performance metrics of interest are obtained. For example, the stationary occupancy probability of cache n is $O^{\mathcal{M}}_{P,n} = \sum_{\mathbf{x}\in\mathcal{E}|x_n=1} \pi(\mathbf{x})$ (the superscript "$\mathcal{M}$" stands for "Markov"). For a line of caches, the hit probability and the miss rate at cache 1 are respectively $H^{\mathcal{M}}_{P,1}(1) = \pi(1,*)$ and $M^{\mathcal{M}}_{R,1} = \lambda_1\pi(0,*)$, while for cache 2 we have

$$H^{\mathcal{M}}_{P,2} = \frac{\lambda_1\pi(0,1,*) + \lambda_2(\pi(0,1,*) + \pi(1,1,*))}{\lambda_1(\pi(0,0,*) + \pi(0,1,*)) + \lambda_2}\ ,\ M^{\mathcal{M}}_{R,2} = \lambda_1\pi(0,0,*) + \lambda_2(\pi(0,0,*) + \pi(1,0,*))$$

where $\pi(i,*) = \sum_{x_2,\ldots,x_N\in\{0,1\}} \pi(i,x_2,\ldots,x_N)$ and $\pi(i,j,*) := \sum_{x_3,\ldots,x_N\in\{0,1\}} \pi(i,j,x_3,\ldots,x_N)$ are the stationary probabilities that cache 1 is in state $i \in \{0,1\}$ and caches $(1,2)$ are in state $(i,j) \in \{0,1\}^2$, respectively. Due to space constraints we omit the general expressions for these quantities for a generic tree of caches. Throughout Section 5.1, we compare the results of our models against the exact ones obtained by studying the Markov process.

Nine caches linear network. This network architecture is chosen for its depth and its small number of leaves. We aim at evaluating the quality of Approximation 4.1 when the depth of the network is large. We consider the tandem of $N = 9$ caches in Figure 8a. At cache n, exogenous requests arrive according to a Poisson process with rate λ_n and TTL is exponentially distributed with mean μ_n^{-1}. We apply Algorithm 2 described in Section 4.4 and compare its prediction to the exact metrics obtained through the analysis of the Markov process $\{\mathbf{x}(t), t \geq 0\}$ introduced in the

previous paragraph. We calculate the absolute relative errors at cache n for the hit probability ($E_{HP,n} := |H^{\mathcal{M}}_{P,n} - H_{P,n}|/H^{\mathcal{M}}_{P,n}$, where $H^{\mathcal{M}}_{P,n}$ is the hit probability obtained from the Markov process analysis), the miss rate ($E_{MR,n}$), and the occupancy probability ($E_{OP,n}$). One thousand different samples for the exogenous request arrival rates and the TTL ones $\{(\lambda_n, \mu_n), n = 1, \ldots, 9\}$ have been selected from the intervals $[0.001, 10]$ and $[0.1, 2]$ respectively. We use the Fourier Amplitude Sensitivity Test (FAST) method [39] to explore the space $[0.001, 10] \times [0.1, 2]$. Figure 9 shows the CCDFs of the relative errors for cache 9. We observe that Approximation 4.1 is very accurate; in 90% of the different parameter settings the relative errors on all metrics of interest are smaller than 10^{-4}.

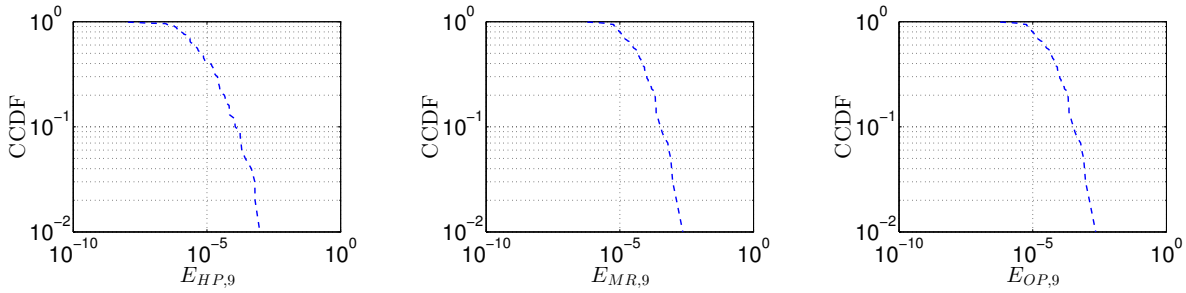


Figure 9: CCDFs of $E_{HP,9}, E_{MR,9}, E_{OP,9}$ for network in Fig. 8a

We then considered a homogeneous scenario where all caches have identical TTL and exogenous arrival rates, i.e. $\mu_n = \mu$ and $\lambda_n = \lambda$, $\forall n$. The relative errors are shown in Figure 10 as a function of the normalized load $\rho = \lambda/\mu$ for $\mu = 0.2$. We observe that the largest error (about 2×10^{-4}) is obtained when arrival and timer rates are comparable (i.e. $\rho \approx 1$).

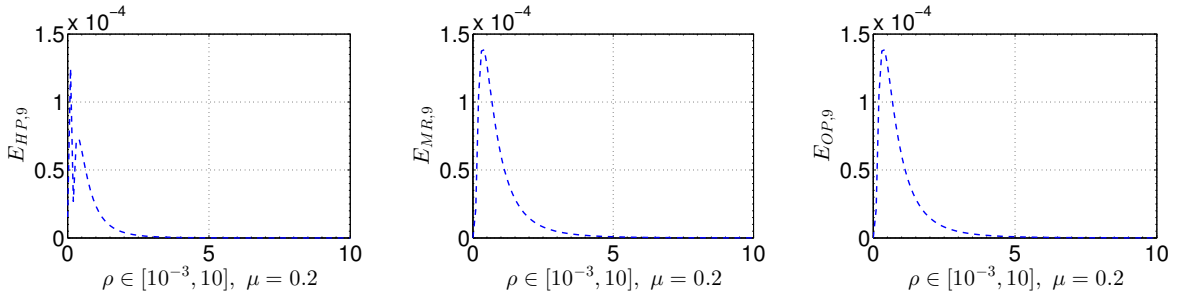


Figure 10: $E_{HP,9}, E_{MR,9}, E_{OP,9}$ for homogeneous network in Fig. 8a ($\lambda_n = \lambda = \rho\mu = \rho\mu_n$)

Twelve caches caterpillar tree. This network consists of three trees (star networks), each with 4 caches, whose roots are connected as in Figure 8b. We choose this network architecture for its large number of leaves and its relative small depth in comparison to the previous linear network. We consider the leaves of each root are identical i.e. they have the same average TTL value and they are fed with Poisson request processes with an identical rate. Again, Algorithm 2 produces exact results for all leaves. As previously, exact results are obtained by considering the Markov process $\{\mathbf{x}(t), t \geq 0\}$ associated to this network. Different request and TTL rates have been selected according to FAST method respectively in the intervals $[0.001, 10]$ and $[0.1, 2]$. We used 4921 samples for each rate. The empirical CCDFs of the relative errors $E_{HP,3}$, $E_{MR,3}$, and $E_{OP,3}$ at the higher level cache are shown in Figure 11.

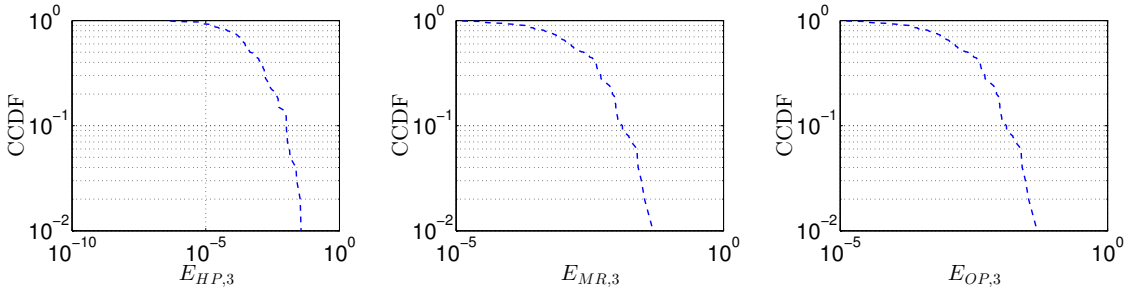


Figure 11: CCDFs of $E_{HP,3}$, $E_{MR,3}$, $E_{OP,3}$ for network in Fig. 8b

The results obtained are analogous to those of the linear cache network in previous paragraph. The relative errors can be larger in this scenario, but they are probably negligible for most of the applications (10^{-2} in 90% of the cases). In this case too, we have also considered the homogeneous scenario where the TTLs at the leaves have the same expected value as the ones at the internal nodes. We observed that the relative errors have the same order of magnitude i.e. less than 10^{-2}.

Nine caches tree network. We consider the tree network of nine caches illustrated in Figure 8c that combines the properties of the previous network samples (i.e. with both a relative large depth and number of leaves). Also in this case, we consider caches are fed by exogenous requests described by Poisson processes and TTLs are exponentially distributed. The request and TTL rates are selected (6649 different samples in total) from the intervals $[0.05, 10]$ and $[0.1, 2]$ respectively using FAST method. Figure 12 shows the CCDFs of the relative errors at the higher level

cache and in 90% of cases they are smaller than 10^{-2}. Thus, Approximation 4.1 is still accurate.

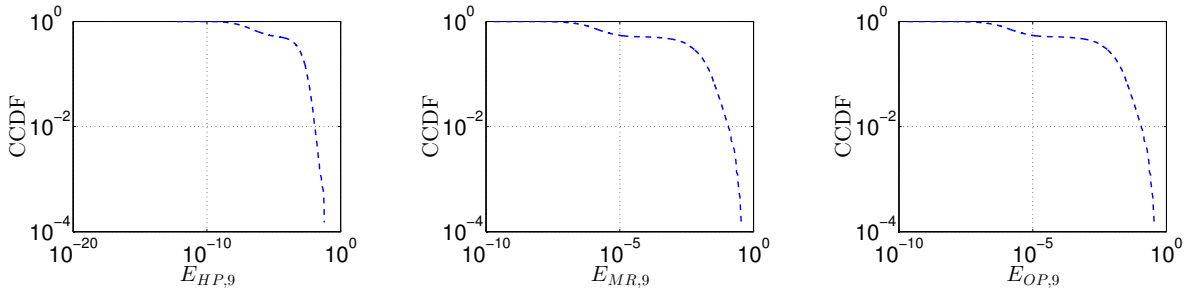


Figure 12: CCDFs of $E_{HP,9}, E_{MR,9}, E_{OP,9}$ for network in Fig.8c

5.2. Poisson traffic and Deterministic timers

When timers are deterministic, we resort to the general procedure in Algorithm 1 presented in Section 4.2. As term of comparison we consider simulation results, given that the network is no longer 'Markovian'.

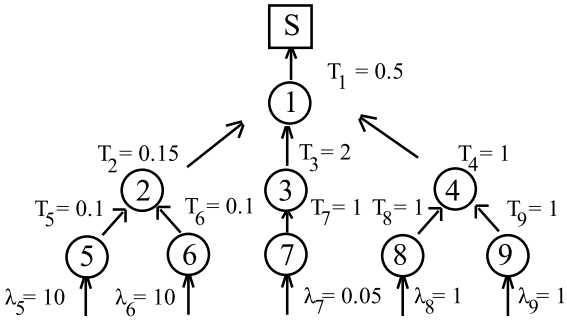


Figure 13: Tree network

Figure 13 shows the settings (topology, request rates and TTL values) of the network.
Algorithm 1 introduces two sources of errors. First, the aggregated request process at a cache is not a renewal process; however, we use Approximation 4.1 and apply the renewal equation (16). Second, (15) and (16) introduce some numerical errors since we need to compute the integrals therein on a finite support. Two parameters determine the size of the numerical error: 1) the time interval (τ) from which the CDF samples are taken, and 2) the time interval between two

consecutive samples (Δ). Clearly the larger τ and the smaller Δ are, the smaller is the numerical error and the larger is the computational cost.

We implemented a MATLAB numerical solver that iteratively determines the CDFs of inter-arrival times at each cache together with the metrics of interest. The integrals appearing in (15) and (16) are approximated by simple sums and for simplicity the same values τ and Δ have been considered for all the CDFs numerical integrations. These parameters are selected as follows: we set the parameter τ to five times the largest expected inter-arrival time in the network; while the parameter Δ is set to one thousandth of the minimum of the TTL values and the expected inter-arrival times of the exogenous request processes.

The relative error of the hit probability is evaluated as $|H_{P,n} - H^{\mathcal{S}}_{P,n}|/H^{\mathcal{S}}_{P,n}$ where $H_{P,n}$ is our estimate and $H^{\mathcal{S}}_{P,n}$ is obtained through simulation. The duration of the simulation is set so that there is a small incertitude on the performance metrics: the 99% confidence interval $[H^{\mathcal{S}}_{P,n} - \epsilon, H^{\mathcal{S}}_{P,n} + \epsilon]$ is such that the ratio $(2\epsilon/H^{S}_{P,n})$ is at most 0.6×10^{-4}. For all the performance metrics at all caches, the relative error of our approach is less than 10^{-2}.

5.3. Renewal/Non-Poisson traffic

In this section, we consider that requests for each data item are generated according to Interrupted Poisson Processes (IPP). IPPs are Renewal processes whose inter-arrival times have a two stage hyper-exponential distribution [21] (then it is a particular Diagonal ME distribution).

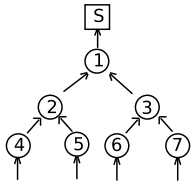


Figure 14: Binary tree network

We evaluate the accuracy of our approach on binary tree networks (like the one in Figure 14) where leaves are fed by request traffic described before and TTLs values are deterministic or drawn from the following *Diagonal ME* TTL distributions: exponential $\big(T(t) = 1 - e^{-\mu t}, t > 0\big)$, 2-stage hypo-exponential $\Big(T(t) = 1 - \frac{\mu_2}{\mu_2 - \mu_1} e^{-\mu_1 t} + \frac{\mu_1}{\mu_1 - \mu_2} e^{-\mu_2 t},\ \mu_1 \neq \mu_2,\ t > 0\Big)$ and 2-stage hyper-exponential $\big(T(t) = 1 - pe^{-\mu_1 t} - (1-p)e^{-\mu_2 t},\ p > 0,\ t > 0\big)$ distributions. Also in this

case we consider simulation results as term of comparison. Our model predictions are provided by Algorithms 1 and 2, respectively for deterministic and (hypo-, hyper-) exponentially distributed TTLs.

Small binary tree. We consider the seven caches binary tree in Figure 14. Relative errors at the higher level cache are displayed in Figure 15. For all performance metrics at all caches of this tree, the relative errors of our approach are less than 2×10^{-3}. This result validates Assumption 4.1 and thus our model in the context of general networks i.e. with non-Poisson arrivals and different TTL distributions.

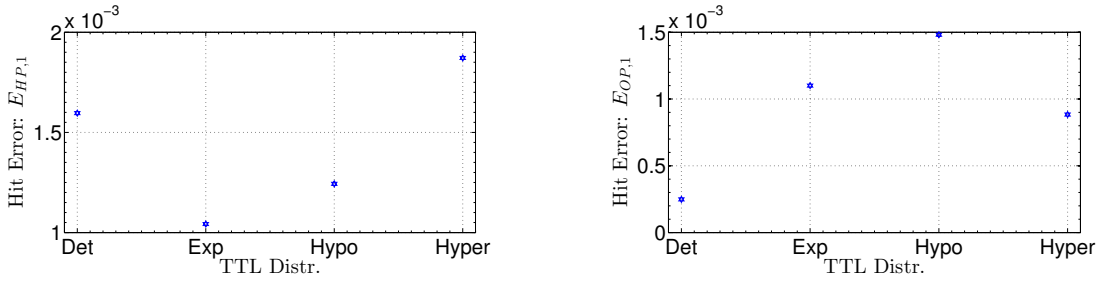


Figure 15: Relative error $E_{HP,1}$ and $E_{OP,1}$ under IPP traffic.

As theoretically proved in Proposition 3.4, Figure 16 confirms that the deterministic TTL is the optimal TTL configuration at the leaves (caches $4-7$) i.e. which maximizes the hit and occupancy probabilities. This observation is not surprising since IPPs are renewal processes with hyper-exponentially distributed inter-arrival times; in fact, it can be easily checked that the hyper-exponential CDF is concave and the observed results follows from Proposition 3.4.

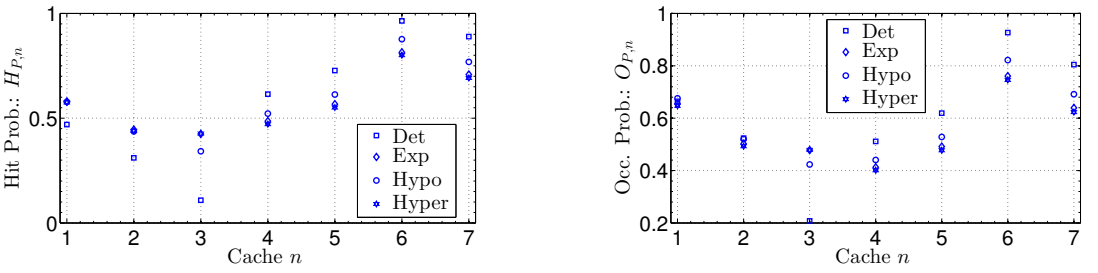


Figure 16: Optimality of the Deterministic TTL at leaves fed by IPP arrivals

Large binary tree. We also investigate the quality of our approximation on larger tree networks (up to 40 caches) where TTLs are constants drawn uniformly at random in the interval $[0.5; 1.5]$, and the exogeneous requests at each cache are described by an IPP. The expected value and the squared coefficient of variation of inter-arrival times are uniformly chosen at random in $[0.05; 2]$ and $[1.5; 2]$ respectively. As shown in Table 3, the relative errors between the event-driven simulations and our analytic approach are of order of 1%. This result provides good insights on the robustness and accuracy of our approach when dealing with large networks.

Type (Degree, Depth, # Caches)	Level l, Cache n	$E_{HP,n}(\%)$	$E_{MR,n}(\%)$	$E_{OP,n}(\%)$
Binary Tree	1, 1	1.059	0.929	0.021
(2, 5, 31)	2, 3	0.406	0.042	0.117
	5, 31	0.075	0.018	0.061
Ternary Tree	1, 1	0.127	0.085	0.134
(3, 4, 40)	2, 3	0.061	0.278	0.124
	4, 40	0.006	0.283	0.759

Table 3: Relative Errors on Performance metrics for large trees

We have shown that Approximation 4.1 leads to very accurate results when exogenous requests are described by renewal process (Poisson and Interrupted Poisson processes) and TTLs have some matrix-exponential distributions or deterministic ones. This lets us think that the superposition of the request arrival processes at every cache is very 'close' to a renewal process at least for all the cases we tested.

6. Computational Cost and Time

In this section we perform a preliminary analysis of the computational cost and time of our approach, and we compare it to other solutions presented in the previous section such as solving a Markov chain (Section 5.1) and event-driven simulations (Sections 5.2 and 5.3).

TTLs with Diagonal ME distribution. We first address the case of a hierarchical tree of Diagonal ME caches introduced in Section 4.4. We consider a tree of N nodes and M internal nodes (i.e. $N - M$ leaves). Since the computational cost for all the metrics is roughly the same, we focus here on the hit probability. In order to calculate the hit probability at one of the nodes labeled $n \in \{1, \ldots, N\}$, say at cache n, we need to:

- calculate the CDF $Z_n(t)$ of inter-arrival times of the aggregated request process in (28). This requires a number of operations proportional to

$$(1+C_n)\prod_{i\in\mathcal{C}(n)\cup\{0\}} K_{i,n} = O\left((1+C_n)\times \tilde{K}_n^{1+C_n}\right),\ \tilde{K}_n = \max_{i\in\mathcal{C}(n)\cup\{0\}} K_{i,n}$$

  where $K_{i,n}$ is the minimal order of the i-th child miss process, $K_{0,n}$ is the minimal order of the exogenous request process, and $C_n = |\mathcal{C}(n)|$ is the number of children of cache n.

- evaluate the LST $Z_n^*(\mu_{n,j})$ in the expression of the hit probability in (21) which requires $K_n \times J_n$ operations where K_n is the minimal order of the aggregated request process (and it is at most equal to $\tilde{K}_n^{1+C_n}$) and J_n is the minimal order of the TTL distribution.

Then, the total cost is

$$\mathcal{K} = O\left(\sum_{n=1}^{N}(1+C_n+J_n)\times \tilde{K}_n^{1+C_n}\right). \tag{30}$$

For linear networks in Figure 5 (case of small maximum degree), the number of children per cache is $C_n = 1$ and there are no exogenous requests at cache $n > 1$. Hence, the total cost is

$$\mathcal{K}_{\texttt{line}} = O\left(NJ\times (K_{0,1}(J+1))^N\right),\ J = \max_{n=1,\ldots,N} J_n \tag{31}$$

For star networks in Figure 6 (case of large maximum degree), the number of children at the root is $N-1$ and the total cost is

$$\mathcal{K}_{\texttt{star}} = O\left(NJK + J\left(K(J+1)\right)^{N-1}\right),\ J = \max_{n=1,\ldots,N} J_n,\ K = \max_{n=1,\ldots,N} K_{0,n} \tag{32}$$

TTLs with exponential distribution. The exponential distribution has the minimal order which is one. Hence, if we consider exponential timers and exogenous requests are described by Poisson processes, we have $K_{0,n} = J_n = 1$ at each cache n. Therefore the costs $\mathcal{K}_{\texttt{line}}$ and $\mathcal{K}_{\texttt{star}}$ are respectively equal to $O(N\times 2^N)$ and $O(N+2^N)$.
We showed in Section 5 that alternative approaches like the Markov chain analysis can provide exact results when the tree is fed by Poisson traffic and the TTLs are exponentially distributed. The size of the state space of the Markov process $\{\mathbf{x}(t), t\geq 0\}$ is 2^N where N is the number of nodes. The cost of determining the steady-state distribution by solving the linear equation

system is $O(2^{3N})$. This is much larger than the cost of our method $O(N2^N)$.
A different approach is to obtain an approximate steady-state distribution of the Markov process using an iterative method. This approach takes advantage of the fact that most of the transition rates are zero. In fact, a state change is triggered by an exogenous request arrival at a cache that does not have the data item or by a timer expiration at a cache with the data item, i.e. from a given state we can only reach other N states. Then the number of non-zero rates is $N \times 2^N$ and each iteration of the method requires $O(N \times 2^N)$ operations. The total cost of the iterative method is then $O(I \times N \times 2^N)$, where I is the number of iterations until termination. The quantity I depends on the spectral gap of the matrix used at each iteration, and also on the required precision. In general, we can expect that $O(I \times N \times 2^N) \ll O(2^{3N})$. Having this inequality, we can say that our method, even in the worst case, is still more convenient than solving the Markov process on linear/star networks, because $O(N2^N) < O(I \times N \times 2^N)$.

TTLs with deterministic distribution. Let us now consider the case of a general tree network with constant TTLs (equal to T). In this case there are no exact solutions to compare our approach with, so we consider simulations as an alternative approach. We perform an asymptotic analysis. A meaningful comparison of the computational costs needs to take also into account the incertitude of the solution: both the simulations and our method can produce a better result if one is willing to afford a higher number of operations. In order to combine these two aspects in our analysis, we consider as metric the product precision times number of operations. Intuitively the larger this product the more expensive is to get a given precision. For the simulations the computational cost is at least proportional to the number of events that are generated, let us denote it by n_E. The incertitude on the final result can be estimated by the amplitude of the confidence interval, that decreases as $1/\sqrt{n_E}$, then the product precision times number of operations is proportional to $\sqrt{n_E}$ for the simulations. In the case of our approach, the most expensive operation is the solution of the renewal equation. If we adopt the same τ and Δ for all the integrals, we need to calculate the value of the CDF of the miss rate ($Y(t)$) in $n_P = \tau/\Delta$ points and then we need to calculate n_P integrals. The integration interval is at most equal to the TTL duration T thanks to (11), then each integral requires a number of operations proportional to $n'_P = T/\Delta$. If the value of τ is selected proportionally to T, then the cost of our method is proportional to n_P^2. A naive implementation of the integral as a sum of the function values leads to an error proportional to the amplitude of the time step and inversely proportional to n'_P or n_P. In conclusion the product precision times the number of operations is proportional to n_P.

Then, for a given precision, our method would require a number of points much larger than the number of events to be considered in the corresponding simulation (at least asymptotically). The comparison would then lead to prefer the simulations at least when small incertitude is required (then large n_E and n_P). In reality integrals can be calculated in more sophisticated ways, for example if we adopt Romberg's method, with a slightly larger computation cost, we can get a precision proportional to n_P^{-2}. In this case the product precision times number of operations is a constant for our method, that should be preferred.

Numerical experiments. We performed some experiments to validate our conclusion based on an asymptotic analysis. First, we consider linear networks of $N = 1, 2, \ldots 9$ exponentially distributed TTL-based caches as described in Figure 8a. We compare the running time of solving the corresponding Markov chain (see Section 5.1) against our Algorithm 2. Figure 17 shows the ratio of the computation times $T^{\mathcal{A}}$ and $T^{\mathcal{M}}$ respectively for our Algorithm 2 and for the Markov chain resolution. Both the solutions have been implemented in MATLAB, in particular the naive function *linsolve* has been used to determine the steady-state distribution of the Markov chain and the Algorithm 2 has been implemented with basic routines. Our algorithm performs faster than the Markov chain resolution specially when the depth of the linear network is large.

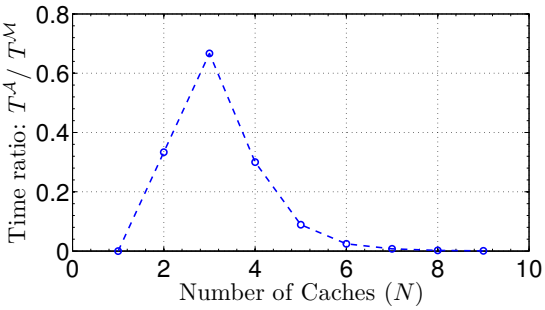


Figure 17: Computation time comparison on linear networks

Second, we evaluate the computational time of the event-driven simulation and our Algorithm 1 on the k-ary trees of Section 5.2 where the TTLs are constants and the request processes are IPPs. $T^{\mathcal{S}}$ and $T^{\mathcal{A}}$ are respectively the time to compute all performance metrics on these large tree networks via event-driven simulations and our analytic methodology in Algorithm 1; they are computed by using the MATLAB routines *tic* and *toc*. Table 4 shows that as the number of caches N increases, our analytic solution is clearly preferable since it is the least time consuming.

Type	Degree	Depth	# Caches, N	$T^{\mathcal{S}}$	$T^{\mathcal{A}}$
Binary Tree	2	5	31	53	88
Ternary Tree	3	4	40	197	129

Table 4: Comparison of computation time on large trees

7. TTL-based model and other policies

We recall that the TTL-based model we presented till now assumes infinite cache capacities. We address issues and practical concerns related to finite capacity constraints.

7.1. Pra-TTL cache: A practical implementation of a TTL-based cache

While the TTL-based model allows an arbitrarily large number of contents in its memory, a real cache will have a finite capacity B. In this section, we consider a possible practical implementation of our TTL-based model that we call *Pra-TTL*. The Pra-TTL cache uses a timer for each content item in the same way as the TTL-based model, but does not discard a content item whose timer has expired as long as some space is available in the memory. If a new content item needs to be stored and the cache is full, the content item to be erased is the one whose timer expired furthest in the past (if any) or the one whose timer will expire soonest. We have compared the performance of the Pra-TTL cache with that of our TTL-based model on a linear network of $N = 5$ caches labeled $n = 1, \ldots, 5$ having the same capacities $B_n = 20$. The requests for each file $f = 1, \ldots, F = 200$ arrive only at the first cache at rate $\lambda_1 = 2.0$ i.e. there are no exogeneous arrivals at caches 2–5. We consider that requests over the set of files follow a Zipf popularity law with parameter $\alpha = 1.2$: i.e. requests for file f are described by a Poisson process with rate $\lambda_{1,f} = \lambda_1 \times \left(1/\sum_g g^{-\alpha}\right)/f^{\alpha}$. Therefore, label f denotes the identifier and the popularity rank of the corresponding file such that file $f = 1$ is the identifier of the most popular file. TTLs of file f at cache n are exponentially distributed with rate $\mu_{n,f} = \mu_n$ such that the total occupancy for the TTL-based model equals the corresponding cache capacity B_n. In other words, μ_n is chosen such that $\sum_{f=1}^{F} O_{P,n,f} = B_n$ where $O_{P,n,f}$ is the occupancy probability of file f at cache n calculted in Proposition 3.2 (i.e. predicted by the model of an infinite TTL-based cache). The hit probability per file f at each cache n is denoted $H_{P,n,f}$ and the aggregate hit probability

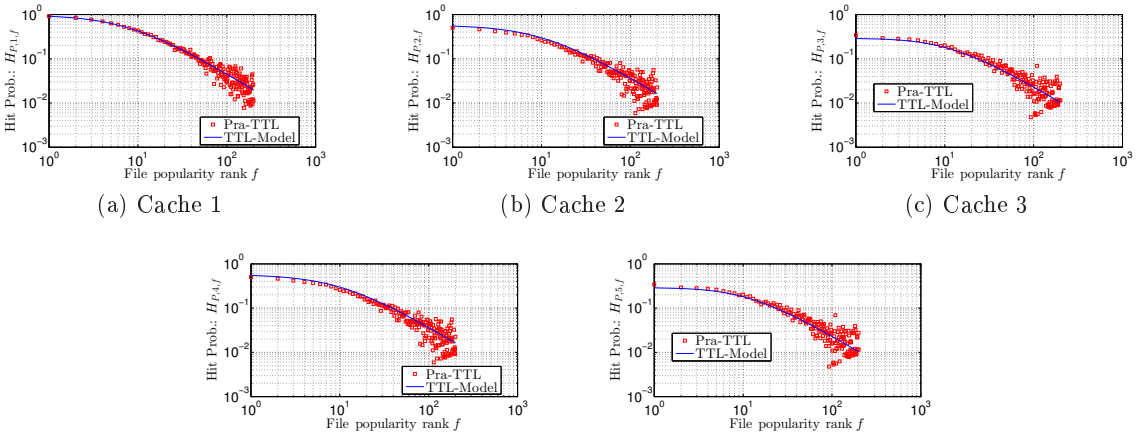
Cache n	Pra-TTL	TTL-Model
1	0.5590	0.5585
2	0.4216	0.4658
3	0.3030	0.2672
4	0.1941	0.1670
5	0.1380	0.1154

Table 5: Aggregated Hit probability at cache n, $H_{P,n,*}$

at cache n is denoted $H_{P,n,*}$. We compute theses performance metrics for both Pra-TTL and TTL-based caches by using the following expression for $H_{P,n,*}$:

$$H_{P,n,*} = \left(\sum_f \Lambda_{n,f} H_{P,n,f} \right) / \Lambda_{n,*}$$

where $\Lambda_{n,f}$ is the total request rate of file f at cache n and $\Lambda_{n,*} = \sum_f \Lambda_{n,f}$. Then, $\Lambda_{n,f}$ is simply the miss rate of file f at cache $n-1$ since the network is linear and there are no exogenous request arrivals at cache n ($\forall\ n > 1$). Table 5 and Figure 18 show that our model (that assume infinite cache size) well predict the performance metrics for Pra-TTL, both those of the aggregate at a cache and those of a specific file respectively.



(a) Cache 1 (b) Cache 2 (c) Cache 3

(d) Cache 4 (e) Cache 5

Figure 18: Hit probability $H_{P,n,f}$ of file f at each cache n: Pra-TTL vs TTL-Model.

These preliminary results suggest that our analysis can be useful to study TTL-based policies under capacity constraints.

7.2. Relationship with other replacement policies

In this section, we establish a link between our TTL-based model and other replacement policies at a single cache. We consider a single cache with capacity B serving F files, where requests are described by independent Poisson processes with rates λ_f for $f = 1, 2, \ldots, F$. We tune the expiration rate μ_f for each file f in order to obtain the same performance metrics of common replacement policies like LRU, FIFO or RND.
We detail the procedure for a single RND cache, but it can be extended to the other policies. Let us denote by π_f the stationary probability that file f is in the RND cache. This distribution has been calculated in [5, 36]. For the exponentially distributed TTL cache, the stationary occupancy probability of the f-th file is given by

$$O_{P,f} = \lambda_f \frac{1 - X_f^*(\mu_f)}{\mu_f},$$

where $X_f^*(s) = \frac{\lambda_f}{\lambda_f + s}$ is the LST of inter-arrival times. If we select $\mu_f = \lambda_f \left(\frac{1}{\pi_f} - 1\right)$, we have $O_{P,f} = \pi_f$, $\forall f$, i.e. the two policies have the same stationary cache occupancy for each file. If we select the same TTL rate μ for all the files it is possible to achieve the same average occupancy at the cache, i.e. $\sum_f \pi_f = \sum_f \frac{\lambda_f}{\lambda_f + \mu} = B$. For each file, the miss process obtained with the exponential TTL-based cache is an accurate description of its miss stream on the RND cache [11]. From the equality of the stationary cache occupancy probabilities, the equality of hit/miss probabilities and rates follows due to the PASTA property since requests are described by Poisson processes.

In this sense, the TTL policy is more general than RND or LRU since it can mimic their behavior and reproduce their performance metrics. While, the exponential TTL cache enables easy calculation we can select other distributions like the deterministic one (see Paragraph *Approximation for LRU caches* in Section 3) in order to better match the CDF of the inter-miss times of a LRU cache as well.

8. Conclusion

In this paper, we introduced a novel Time-To-Live (TTL) based replacement policy for cache networks in general and the content-routers of ICN architectures in particular. We developed a set of building blocks for the performance evaluation of theses TTL-based cache networks through renewal arguments. We characterized a class of networks for which we provided the exact performance metrics: this class contains linear and star tree networks. We also provided a recursive and approximate procedure to study arbitrary hierarchical networks. We showed that our theoretic model predicts remarkably well the performance metrics with relative errors less than 1%. We formally proved that deterministic TTLs are optimal when the inter-arrival times have a concave CDF. Our approach is promising since it appears as a unifying framework to accurately analyze a richer class of networks also with heterogeneous policies deployed at different caches. We have also demonstrated that our TTL-based model can be implemented under capacity constraints. Ongoing research is investigating approximate TTL-based model for finite capacity caches running the LRU, FIFO and Random replacement policies. We also aim at considering the case of correlated requests modeled by semi-Markov processes.

Acknowledgements

This research has been supported by Orange Labs in the framework of the contract "Performance models for CCN architecture." We thank in particular Orange Labs researchers Bruno Kauffmann, Luca Muscariello and Alain Simonian for the stimulating discussions and for the valuable feedback. This material is also based upon work supported by the National Science Foundation under Grant No. CNS-1040781.

Appendix

Optimality of a deterministic TTL-cache

In this appendix, we obtain the TTL distribution that maximizes/minimizes our metrics of interest (i.e. the hit probability H_P and the occupancy probability O_P) when the mean TTL value $D = \mathbb{E}[T]$ is known.

Lemma 8.1 (Convex ordering)**.** *If D and T are respectively constant and random TTLs such that $\mathbb{E}[T] = D$, then the following relation holds*

$$D \leq_{\texttt{cx}} T \tag{33}$$

where $\leq_{\texttt{cx}}$ is the convex ordering.

Proof. The definition of convex ordering of random variables T_1 and T_2 says $T_1 \leq_{\texttt{cx}} T_2$ if and only if $\mathbb{E}[\phi(T_1)] \leq \mathbb{E}[\phi(T_2)]$ where $\phi(.)$ is a convex function. We shall show that this convex ordering holds for any random TTL T and constant TTL D such that $\mathbb{E}[T] = D$ in order to prove the lemma. For any random TTL $T \geq 0$ and any convex function $\phi(.)$, we have thanks to Jensen's inequality :

$$\mathbb{E}[\phi(T)] \geq \phi(\mathbb{E}[T]) = \phi(D) = \mathbb{E}[\phi(D)]$$

The last equality follows from the fact that $\phi(D)$ is a constant. ◇

Proposition 8.1 (Optimality of a deterministic TTL cache)**.** *Given the expected TTL value $D = \mathbb{E}[T]$ and the CDF $X(t)$ of inter-arrival times, the occupancy O_P is maximized when the TTL is deterministic and equal to D. Moreover, if $X(t)$ is a concave function then the hit probability H_P is maximized too.*

Proof. We assume that the TTLs $\{T_n\}_{n\geq 1}$ are sampled from a general distribution $T(t)$ such that $\mathbb{E}[T] = D$. Observe that the occupancy probability $O_P(T)$ and hit probability $H_P(T)$ are functions of the timer T and can be written as

$$O_P(T) = \lambda \mathbb{E}\left[\phi(T)\right] \ , \ H_P(T) = \mathbb{E}[X(T)]$$

where $X(x)$ is the CDF of X and $\phi(t) = \int_0^t (1 - X(x))\mathrm{d}x$. The second derivative of $\phi(t)$ is $\phi''(t) = -X'(t) \leq 0$ because $X'(t)$ is a probability density function; hence, $\phi(t)$ is a concave function for any $X(x)$. Then by applying Lemma 8.1, it follows that $O_P(T) \leq O_P(D)$ for any timer T such that $\mathbb{E}[T] = D$. Meanwhile, if $X(x)$ is concave (resp. convex), Lemma 8.1 states that $H_P(D) \geq H_P(T)$ (resp. $H_P(D) \leq H_P(T)$). ◇

We note that if the request process is a Poisson process, the occupancy O_P and the hit probability H_P are equals and theses metrics are maximized when the TTL is deterministic.

References

[1] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher and B. Ohlman, "A survey of information-centric networking", *IEEE Communications Magazine*, Vol. 50, No. 7, pp. 26-36, Jul. 2012.

[2] F. Baccelli and P. Brémaud, *Elements of Queueing Theory. Palm Martingale Calculus and Stochastic Recurrences.* Applications of Mathematics, Stochastic Modelling and Applied Probability, Vol. 26, Springer, 2nd Edition, 2003.

[3] O. Bahat and A. M. Makowski, "Measuring consistency in TTL-based caches", *Performance Evaluation*,Vol., 62, Issues 1-4, pp. 439-455, Oct. 2005.

[4] J. R. Bitner, "Heuristics that monotonically organize data structures", *SIAM J. Computing*, **8**, pp. 82-110, 1979.

[5] P. J. Burville and J. F. C. Kingman, "On a model for storage and search", *J. of Applied Probability*, **10**, pp. 697-701, 1973.

[6] J. Mc Cabe, "On serial files with relocable records", *Operations Research*, **13**, pp. 609-618, 1965.

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

[8] N. Choungmo Fofack, P. Nain, G. Neglia and D. Towsley, "Analysis of TTL-based cache networks", *INRIA Research Report RR-7883*, 2012.

[9] G. Carofiglio, M. Gallo, L. Muscariello and D. Perino, "Modeling data transfer in content-centric networking", *Proc. 23rd International Teletraffic Congress* (ITC 23), San Francisco, CA, USA, Sep. 6-8, 2011.

[10] H. Che, Y. Tung and Z. Wang, "Hierarchical Web caching systems: modeling, design and experimental results", *IEEE J. on Selected Areas in Communications*, Vol. 20, No. 7, pp. 1305-1314, Sep. 2002.

[11] V. Martina, M. Garetto and E. Leonardi, "A unified approach to the performance analysis of caching systems", *http://arxiv.org/abs/1307.6702*, Sep. 10 2013.

[12] E. Çinlar, *Introduction to Stochastic Processes.* Prentice Hall, 1975.

[13] E. G. Coffman Jr. and P. Jelenkovic, "Performance of the move-to-front algorithm with Markov-modulated request sequences", *Operations Research Letters*, **25**, pp. 109-118, 1999.

[14] E. Cohen, E. Halperin and H. Kaplan, "Performance aspects of distributed caches using TTL-based consistency", *Theoretical Computer Science*, pp 73-96, Feb. 2005.

[15] A. Dan and D. Towsley, "An approximate analysis of the LRU and FIFO buffer replacement schemes", *Proc. ACM Sigmetrics 1990*, pp. 143-152, Boulder, CO, USA, May 22-25, 1990.

[16] R. P. Dobrow and J. A. Fill, "The move-to-front rule for self-organizing lists with Markov dependent requests", *Discrete Probability and Algorithms*, IMA Volumes in Mathematics and its Applications, D. Aldous, P. Diaconis, J. Spencer, and J. M. Steele (Eds), **72**, pp. 57-80, Springer-Verlag, 1995.

[17] A. Feldmann and W. Whitt, "Fitting mixtures of exponentials to long-tail distributions to analyze network performance models", *Proc. IEEE Infocom 1997*, Kobe, Japan, Apr. 7-11, 1997.

[18] W. Feller, *An Introduction to Probability Theory and its Applications.* Vol. 2. John Wiley & Sons Ltd, New York, 2nd Edition, 1971.

[19] J. A. Fill, "Limits and rate of convergence for the distribution of search cost under the move-to-front rule", *Theoretical Computer Science*, **176**, pp. 185-206, 1996.

[20] J. A. Fill and L. Holst, "On the distribution of search cost for the move-to-front rule", *Random Structures Algorithms*, Vol 8, No 3, pp. 179-186, 1996.

[21] W. Fischer and K. Meier-Hellstern, "The Markov-modulated Poisson process (MMPP) cookbook", *Performance Evaluation*, Vol. 18, pp. 149-171, Jan. 1991.

[22] P. Flajolet, D. Gardy and L. Thimonier, "Birthday paradox, coupon collectors, caching algorithms and self-organizing search", *Discrete Applied Mathematics*, **39**, pp. 207-229, 1992.

[23] C. Fricker, P. Robert and J. Roberts, "A versatile and accurate approximation for LRU cache performance", *http://arxiv.org/abs/1202.3974v1*, Feb. 2012.

[24] E. Gelenbe, "A unified approach to the evaluation of a class of replacement algorithms", *IEEE Trans. on Computers*, C-22(6), 1973.

[25] J. Gray and F. Putzolu, "The 5-minute rule for trading memory for disc accesses and the 5-byte rule for trading memory for CPU time", *Technical Report 86.1*, TandemComputers, PN 87615, May 1985. Available at `http://www.hpl.hp.com/techreports/tandem/TR-86.1.pdf`.

[26] Q-M. He and H. Zhang "On matrix exponential distributions", *Adv. in Applied Probability*, **39**, pp. 271-292, 2007.

[27] W. J. Hendricks, "The stationary distribution of an interesting Markov chain", *J. of Applied Probability*, **9**, pp. 231-233, 1972.

[28] V. Jacobson, D. K. Smetters, J. D. Thorntorn, M. Plass, N. Briggs and R. L. Braynard, "Networking named content", *Proc. ACM CoNEXT 2009*, Rome, Italy, Dec. 1-4, 2009.

[29] P. Jelenkovic, "Asymptotic approximation of the move-to-front search cost distribution and least-recently used caching fault probabilities", *The Annals of Probability*, Vol. 9, No. 2, pp. 430-464, 1999.

[30] P. Jelenkovic and A. Radovanović, "Least-recently used caching with dependent requests", *Theoretical Computer Science*, **326**, pp. 293-327, 2004.

[31] P. Jelenkovic, A. Radovanović and M. Squillante, "Critical sizing of LRU caches with dependent requests", *J. of Applied Probability*, Vol. 43, No. 4, pp. 1013-1027, Dec. 2006.

[32] J. Jung, E. Sit, H. Balakrishnan and R. Morris, "DNS performance and the effectiveness of caching", *Proc. ACM Sigcomm Workshop on Internet Measurement (IMW '01)*, New York, NY, USA, Nov. 1-2, 2001.

[33] J. Jung, A. W. Berger and H. Balakrishnan, "Modeling TTL-based Internet caches", *Proc. IEEE Infocom 2003*, San Francisco, CA, USA, Mar. 30 - Apr. 3, 2003.

[34] S. Karlin, *Total Positivity.* Vol. 1, Stanford University Press, 1968.

[35] J. Kawash, "Consistency models for Internet caching", *in Proc. of the Winter International Symposium on Information and Communication Technology*, pp 161-166, Jan. 2004.

[36] W. F. King, "Analysis of demand paging algorithm", *Information Processing*, Vol. 71, pp. 485-490, 1972.

[37] T. Lai and H. Robbins, "Asymptotically efficient adaptive allocation rules", *Adv. in Applied Mathematics*, Vol. 6, pp.4-22, 1985.

[38] N. Laoutaris, S. Syntila and I. Stavrakakis, "Meta algorithms for hierarchical Web caches", *23rd IEEE International Performance Computing and Communications Conference* (IPCCC 2004), Phoenix, Arizona, Apr. 15-17, 2004.

[39] I. Lassoued, A. Krifa, C. Barakat and K. Avrachenkov, "Network-wide monitoring through self-configuring adapative system", *Proc. IEEE Infocom 2011*, Shanghai, China, Apr. 10-15, 2011.

[40] A. J. Lawrence, "Dependency of intervals between events in superposition processes", *J. of the Royal Statistical Society*, Series B, Vol. 35, No. 2, pp. 306-315, 1973.

[41] B. L. Nelson and I. Gerhardt, "On the capturing dependence in point processes: Matching moments and other techniques", *Working Paper*, Jan. 2010.

[42] A. D. Polyanin and A. V. Manzhirov, *Handbook of Integral Equations.* CRC Press, Boca Raton, 1st Edition, 1998.

[43] E. J. Rosensweig, J. Kurose and D. Towsley, "Approximate models for general cache networks", *Proc. IEEE Infocom 2010*, San Diego, CA, USA, Mar. 15-19, 2010.

[44] 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*, Vol. 36, pp. 315-327, 2002.

[45] A. Simonian, M. Gallo, B. Kauffmann, L. Muscariello and C. Tanguy, "Performance of the random replacement policy for networks of caches" *Proc. of ACM Sigmetrics/Performance*, London, UK, Jun. 11-15, 2012.

[46] X. Tang, J. Xu and W. Lee, "Analysis of TTL-based consistency in unstructured peer-to-peer networks", *IEEE Trans. on Parallel and Distributed Systems*, Vol. 19, No. 12, Dec. 2008.

[47] W. Whitt, "Approximating a point process by a renewal process, I: Two basic methods", *Operations Research*, Vol. 30, No. 1, Jan-Feb. 1982.