An approximate analysis of heterogeneous and general cache networks

Nicaise Choungmo Fofack, Don Towsley, Misha Badov, Mostafa Dehghan, Dennis L. Goeckel

An approximate analysis of heterogeneous and general cache networks.

Nicaise Choungmo Fofack, Don Towsley , Misha Badov, Mostafa Dehghan, Dennis L. Goeckel

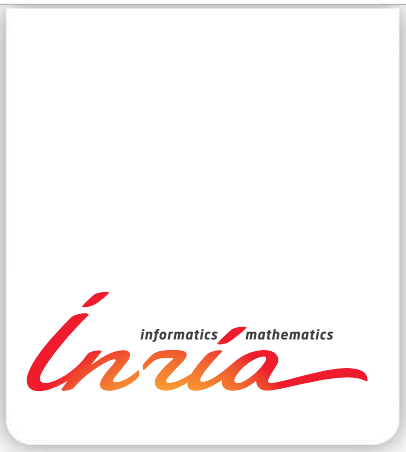
RESEARCH REPORT

N° 8516

April 8 2014

Project-Team Maestro

ISSN 0249-6399 ISRN INRIA/RR--8516--FR+ENG



An approximate analysis of heterogeneous and general cache networks.

Nicaise Choungmo Fofack *, Don Towsley †, Misha Badov†, Mostafa Dehghan†, Dennis L. Goeckel ‡

Project-Team Maestro

Research Report n° 8516 — April 8 2014 — 33 pages

Abstract: In this paper, we propose approximate models to assess the performance of a cache network with arbitrary topology where nodes run the Least Recently Used (LRU), First-In First-Out (FIFO), or Random (RND) replacement policies on arbitrary size caches. Our model takes advantage of the notions of *cache characteristic time* and *Time-To-Live (TTL)-based cache* to develop a unified framework for approximating metrics of interest of interconnected caches. Our approach is validated through event-driven simulations; and when possible, compared to the existing *a-NET* model [23].

Key-words: On-demand cache, Cache networks, Approximate analysis, Least Recently Used (LRU), Random replacement (RND), First-In First-Out (FIFO), Time-To-Live (TTL), Cache Characteristic Time, Content-oriented Networks.

* Email: {nicaise.choungmo_fofack, philippe.nain}@inria.fr. Postal address: Inria, B.P. 93, 06902 Sophia Antipolis, Cedex, France.

† Email: {towsley, mbadov, mdehghan}@cs.umass.edu. Postal address: School of Computer Science, University of Massachusetts at Amherst, 181 Governors Dr Amherst, MA 01003-9264, USA.

‡ Email: goeckel@ecs.umass.edu. Postal address: Department of Electrical and Computer Engineering, University of Massachusetts at Amherst, 181 Governors Dr Amherst, MA 01003-9264, USA.

RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Analyse approximative d'un réseau hétérogène et général de caches.

Résumé : Dans ce travail, nous proposons des modèles approximatifs pour évaluer les performances d'un réseau de caches ayant une topologie arbitraire où les noeuds exécutent les politiques Least Recently Used (LRU), First In First Out (FIFO), ou Random replacement (RND) sur des caches de taille quelconque. Notre modèle tire parti des notions de *temps caractéristique* d'un cache et des modèles *Time-To-Live (TTL)* de cache pour développer une approche unifiée pour l'approximation des métriques de performance sur des caches interconnectés. Notre approche est validée par des simulations événementielles; et, si possible, comparée au modèle existant *a-NET* [23].

Mots-clés : Cache-à-la demande, Réseaux de caches, Analyse approximative, Moins récemment utilisé (LRU), Remplacement alÃ©atoire (RND), Premier arrivé premier servi (FIFO), Temps caractéristique, Durée de vie (TTL), Réseaux orientés contenu.

1 Introduction

Over the past years, popular cache management algorithms such as Least Recently Used (LRU), First-In First-Out (FIFO), and Random (RND) have received significant attention. Nowadays, interest has shifted from caching systems in isolation to interconnected caches. The benefits of the latter approach come from storing contents in caches that are universally deployed or distributed across the network. This trend is mostly driven by the recent development of content-oriented technologies such as Domain name System (DNS), Content Distribution Networks (CDNs), and the recent concept of Information-Centric Networking (ICN) [1]. The purpose is to adapt the network architecture to accommodate the current content usage patterns (Video-on Demand, User Generated Contents, Web browsing, etc.) with the potential to reduce congestion, improve content delivery speed as networks increase in size.

The latter performance goals can be achieved in different ways. We focus on content-oriented networking technologies like the DNS [15], Web [4], CDNs [24], and ICNs [1] which allow *on-demand* caching through deployment of caches at various locations. It is briefly described as follows: when data is first requested, it is temporarily stored at some nodes; and thus, subsequent demands may be served directly from these local copies. Therefore, users may experience a better quality of service. Obviously, the benefits of such in-network caching choices are highly tied to the design of the underlying cache networks. The latter task can be significantly challenging; in fact, the multi-cache systems that rise from content-oriented architectures are irregular and heterogeneous. More precisely, these networks may have arbitrary topologies and their nodes may present different characteristics (cache replacement policies, capacities, routing tables, etc.).

Although considerable work has focused on studying isolated caches (see [10] and references therein), there have been few attempts to model cache networks. Note that exact analysis of general cache networks is extremely difficult, and requires vast computational resources. On isolated cache the complexity of an exact analysis grows exponentially with the cache size C and the number of files N. Hence, approximate models are derived in [20], [4, 19], and [23] respectively for linear, tree, and graph-based networks of LRU caches. Meanwhile [12] addressed tree-based networks of RND caches and provided preliminary results on a tandem of two caches running LRU and RND replacement policies. Polytree-based Time-To-Live (TTL) cache networks are studied in [6, 21, 3, 10]. However, the general problem of arbitrary topologies and heterogeneous cache networks is still poorly understood.

Unfortunately, existing models for networks of caches are limited by one of the following:

(i) the network topology, which is generally assumed to be hierarchical or tree-based [4, 19, 20, 12, 17];

(ii) the routing/forwarding schemes, which are restricted to the case where requests flow in the same direction, e.g. from-children-to-parents forwarding schema of tree network;

(iii) the cache replacement policies, which are required to be identical across all caches;

(iv) the Independent Reference Model (IRM) or Poisson approximation [9], which is inaccurate [14] and commonly used to represent request streams ([23] reports relative errors of 16%); and finally

(v) the computational complexity due to exact calculation [6, 21, 20], which can be significant especially for large networks.

Clearly, there is a lack of tools for performance evaluation of general and heterogeneous cache networks that can help engineers to gain insights into how interconnected caches perform, to test several network configurations, and to design their own network without investing too much effort and resources on test-beds. We address these five issues and derive approximate methodologies (and algorithms) by leveraging the notion of *cache characteristic time* for isolated caches [4, 19], techniques of approximating general request processes by *renewal processes* [27, 2], exact theoretic results on *performance metrics calculation* of TTL-based caches [21, 5, 10], and modelling approaches of complex *routing* schemes [25, 13]. Our methodology is validated through extensive event-driven simulations and compared, when possible, to *a-NET* model [23] which is an existing approximation methodology for general and homogeneous cache networks. To the best of our knowledge, this is the first attempt to study cache networks in a "truly" general sense.

The rest of this paper is organized as follows. Section 2 presents relevant work under which our approach relies. It also shows the limitations of existing methodologies. We introduce our network model, assumptions and solution schema in Section 3. Section 4 describes the approximate models and algorithms for studying single LRU, FIFO, and RND caches. We also report some misconceptions on characteristic times found in the literature [19]. In Section 5, we focus on performance analysis of cache networks fed by independent and identically distributed (i.i.d.) requests and we present challenges in terms of network primitives that we address by translating the latter into well-known operations on request processes. We also describe how we can account for correlated request streams. Our modeling approach is evaluated in Section 6 and our findings are summarized in Section 7.

2 Related Work

The closest work, methodologically speaking, to ours is the 2010 paper by Rosensweig, Kurose and Towsley [23]. The authors provide an in-depth analysis of sources of inaccuracies but also a simplified methodology for performance evaluation of general LRU cache networks. Their approach is built on top of the LRU cache approximation developed by Dan and Towsley [8], which has a complexity of order of $O(NC)$ where N is the number of files and C the cache size. Besides the complexity which can be significant for typical content-oriented networks (i.e. large values of N and C), [23] has identified three potential sources of prediction error of their analysis of general and homogeneous networks of LRU caches: (e_1) the inaccurate performance metrics calculation of [8], (e_2) the violation of the IRM (or Poisson) assumption on the miss streams of LRU caches (already shown by Jelenkovic and Kang [14]), and (e_3) the inaccurate characterization of the aggregated request process that feed a cache within a network (this process may result from the superposition of miss streams from other caches and exogenous requests).

Che et *al.* [4] proposed an alternative approach to that of Dan and Towsley [8] for the calculation of performance metrics of LRU caches. Their model is based on the *cache characteristic time* and was experimentally proved to be very accurate on hierarchical (more precisely on two-level tree-based) cache networks. To some extent, the *characteristic time approximation* (CTA) for a single LRU cache [4] addresses the source of prediction error (e_1) with a precision comparable to that of [23]. However, the complexity of the cache performance calculation using the CTA is not well established as the iterative procedure of [8, 23]. Moreover, the network model in [4] still suffers from: (e_2) i.e. the characterization of the miss streams of LRU caches, the large complexity of the exact characterization [18] of the aggregated request process at the root, the limitation of tree cache networks, and the strong dependence on Poisson request streams at leaves.

Martina et *al.* [19] experimentally extended the results of Che et *al.* [4] in three directions. First, their simulations showed that the CTA can be applied to other cache replacement policies such as LRU, FIFO, RND and variants. Second their numerical results showed that the Poisson request streams assumption at leaves can be relaxed and non-IRM models such as renewal request processes can be safely considered. Third their analysis is carried out on arbitrary hierarchical (tree) cache networks. However, their network model did not successfully address errors (e_2) and (e_3) since miss streams of requests and aggregated request processes are approximated by Poisson processes. Also, they state that a tandem of two caches "*with requests flowing in the same direction*" suffices to analyze general cache networks. We shall see in Section 5 that several operations such as *aggregating, splitting of request streams*, and also "*requests flowing in opposite direction*" cannot be handle or trivially derived from their tandem of two caches example.

[6, 21, 10] and [3] studied TTL-based cache networks and provided exact analysis under renewal and Markov arrival request processes respectively. Under the CTA, authors of [10, 5] showed that TTL-based models can accurately describe the behavior (hit/occupancy probabilities and miss streams) of LRU, FIFO and RND caches. This claim is partially supported by recent results of Melazzi et *al.* [20] the on performance analysis of linear LRU cache networks. Simulations in [10] showed that performance metrics at each node can be obtained with absolute relative errors less than 5%. Hence, by combining the CTA and TTL-based models, one can directly tackle the three sources of prediction errors (e_1), (e_2) and (e_3) on general and heterogeneous networks of LRU, FIFO and RND caches. However, the exact analysis carried out in [6, 21, 10, 20] is computationally expensive since it requires the evaluation of integrals/convolutions over infinite supports and/or the resolution of integral equations. This is definitely an handicap for the analysis of large and arbitrary cache networks.

In this paper, we develop an approximate methodology for general and heterogeneous cache networks that also addresses the numerical complexity observed in [10, 20] while providing a comparable level of accuracy. We present our approach in the next section and it is worth noting that our approach can be easily extended to include other replacement policies such as q-LRU and Least Frequently Used (LFU) as introduced in [19].

3 Model and assumptions

In this section, we first describe the system of interest and formally define the problem that we try to solve. Then we explain the approach we take to tackle the problem; and finally, we provide preliminary and new results for isolated caches.

3.1 Network model and problem statement

Let $\mathcal{G} = (V, E)$ be the graph representing a general and heterogeneous cache network, $V = \{v_1, \ldots, v_N\}$ the set of caches, and $E \subset V \times V$ the set of connections between caches. Additionally, the file catalog is denoted by $\mathcal{F} = \{f_1, \ldots, f_K\}$. Each file is stored permanently at one or more public servers attached to nodes in the network.

Requests arrive exogenously and directly from users to some of the nodes $\{v_n \in V\}$ in this system. When a request for file f_i arrives at a cache, it generates a cache hit if the file is located in the cache and a miss otherwise. In the latter case, the request is forwarded to other caches in the network based on the routing table at each cache, until the file is located in a cache or at the server storing the file. Then the file is forwarded along the reverse path taken by the request, and stored at each cache along the way; this network caching strategy is also known as *Leave Copy Everywhere* (LCE) by Laoutaris, Syntila and Stavrakakis [17].

If the cache is full when a miss occurs, one of the files in the cache is selected based on an eviction policy to make room for the new file. In this work, we consider that caches may run one of the three most popular replacement policies, namely, LRU, FIFO, and RND. However, the model can be easily extended to include other variant replacement policies studied in [19]. Following common practice, we assume that all files have the same size (see [23] and references therein); otherwise, they can be divided in small chunks of identical size (see [11]). Hence we express the cache size in terms of the number of files/chunks it can hold at any given moment. As commonly agreed [8, 4, 23], we also assume that the request processing/forwarding times and the file download times after a cache miss are significantly smaller than the inter-request timescale. This assumption has been validated on real DNS traces in [21]. Thus, once a cache miss occurs, the file is instantaneously available in the cache.

3.2 Processes at hand

We denote by $\mathcal{R}_{n,i} = \{t_k(n,i)\}_{k\geq 0}$ the overall request process of file f_i at cache n where $t_k(n,i)$ is the arrival instant of the $k+1$-th request, $\Lambda_{n,i}$ the intensity of the process $\mathcal{R}_{n,i}$, and $\lambda_{n,i}$ the rate of exogenous arrival at cache v_n. In the remainder of this paper, the following statement holds unless otherwise specified.

Assumption 1 (Renewal process) *Exogenous requests and aggregated streams of requests are described by a renewal process.*

We denote by $X_{n,i}$ the generic inter-arrival time of the process $\mathcal{R}_{n,i}$, $F_{n,i}(t) = P(X_{n,i} < t)$ its Cumulative Distribution Function (CDF), $\hat{F}_{n,i}(t)$ the CDF of its survival time (or forward recurrence time), $M_{n,i}(t)$ the Renewal Function associated with $F_{n,i}(t)$, and $F^*_{n,i}(s) = E[e^{-sX_{n,i}}]$ its Laplace-Stieltjes Transform (LST) for all $t \geq 0$ and $s \geq 0$. We refer to classical references on *renewal theory*, such as the book by Cox [7], for detailed definitions of theses quantities.

Assumption 1 does not hold in general especially for aggregated processes unless each request stream being aggregated is a Poisson process. However, renewal processes have been shown to be sufficiently general to accurately describe real traces (see [15, 21]). Here, we aim at providing simple and accurate approximations of $\{\mathcal{R}_{n,i}\}$ based on a *minimal available information* that

engineers could easily measure or estimate at edge nodes of the network. We choose the *frequency* (or *request rate*) and the *burtiness* (or *coefficient of variation* of exogenous inter-request times) as inputs of our process model. These parameters are general enough to handle per-file popularity and temporal locality in request processes [19, 26, 10, 20].

Whitt's approximation of general point processes [27] In this paper, exogenous, miss and aggregated request processes will be approximated by renewal processes having hyper-exponential or shifted-exponential CDF for inter-arrival times introduced by Whitt [27]. The characteristics of these approximate renewal processes will be intensively used and the reader should refer to Appendix A if needed.
Thanks to Whitt's approximations, our task of characterization of request streams reduces to determine the frequency and the burstiness of incoming and miss request streams at each node of the network. Our approach is detailed in the next section.

3.3 Solution schema and contributions

In this section, we describe our procedure for a single cache. The latter will be iterated on cache networks. Our approach relies on the following building blocks.

(b_1) **The Arrival Process Approximation (APA)** This approximation is used to provide a simplistic description of request processes of each file at each cache of the network. It takes advantage of Assumption 1 and well-known approximation procedures [27, 2] to describe a general point process. The method used here is known as *moment matching* [22] since it requires knowledge of the first two moments of the inter-request times.

(b_2) **The Characteristic Time Approximation (CTA)** The characteristic time T_i of file f_i of a single cache is defined as the maximum inter-request time of that file which leads to cache hit. This notion was initially introduced by Che et *al.* [4] for LRU caches and later extended by Martina et *al.* [19] to other replacement policies such as RND, FIFO and variants under Assumption 1. It was experimentally [4, 16, 19] verified that $E[T_i] \approx T$, $\forall i$ especially when the rate Λ_i is negligible [11] in comparison to the total request rate $\Lambda = \sum_{i=1}^{K} \Lambda_i$. Simulations in [4, 19] confirmed that LRU and FIFO caches have a deterministic characteristic time, while that of RND caches is exponentially distributed. Hence, this allows us to study LRU, RND and FIFO caches through their corresponding TTL-based models where files are decoupled such that the analysis we derive for one file applies for the others in the same fashion [21, 10].

(b_3) **The Performance metrics and Miss process Characterization (PMC)** We rely on exact and closed-form formulas of performance metrics and miss streams of TTL-based caches derived in [21, 10] under Assumption 1. We recall that the miss process of a TTL cache was shown to be a renewal process when Assumption 1 holds. Hence, we only need to compute exactly the first two moments of the cache miss process.

(b_4) **The Routing of Request Streams (RRS)** We assume that requests of each file are routed on an arbitrary feed-forward network built on top of the network topology from points/nodes where requests occur to one or many servers that permanently hold the file.

Combining CTA and PMC, the TTL-based model of LRU caches initializes the timer of a file when cache misses occur and later resets this timer at each cache hit as shown in Figure 1; while that of FIFO caches sets the timer only when a cache miss occurs as depicted in Figure 2.

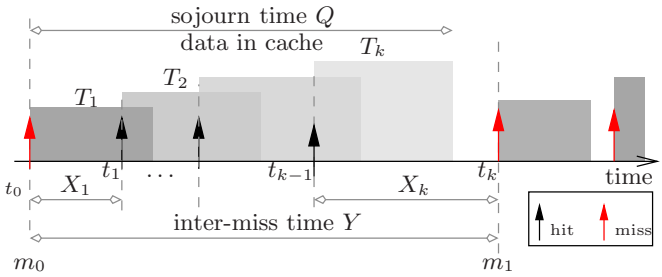


Figure 1: TTL-model of single file in LRU cache

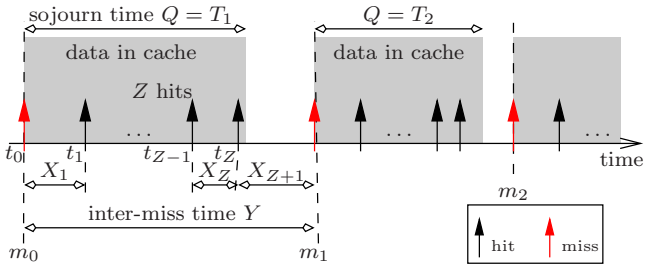


Figure 2: TTL-model of single file in FIFO cache

This is explained by the LRU (resp. FIFO) algorithm which inserts a requested file at the head of the cache (resp. if the file is not in the a cache). However, a cache hit does not change the state of a FIFO cache. According to these different TTL-models, LRU and FIFO caches can be analyzed via the class of *TTL-resetting* caches [6] and *TTL-non-resetting* [21] caches respectively. RND caches can be modeled by either of these classes of TTL-based policies. This is due to the memoryless property of the exponential TTL distribution of TTL-based models of RND caches. In the following, the terms *characteristic time* and *TTL* will be used interchangeably.

3.4 General results on single cache under Assumption 1

In this section, we define the metrics of interest and we establish results which hold for any cache policy under Assumption 1. From now on we omit the subscript n that refers to the cache label in all the quantities we introduced, or the ones we shall define later.

We consider the TTL T_i of a file f_i as defined in the previous section and we denote by Q_i the expected time that file f_i resides/sojourns in the cache (see the notation *data in the cache* in Figures 1 and 2). Also, Λ_i, $H_{P,i}$ and $O_{P,i}$ denote the request rate, the hit probability (i.e. probability that an arriving request finds file f_i in the cache), and the occupancy (i.e. stationary probability that the file f_i is in the cache at any time instant) for file i, respectively. For quick reference, notations are summarized in Table 1.

The next result links metrics of interest and the sojourn time of a file in the cache.

Lemma 1 (Little's Law) *Under Assumption 1, the following relations hold:*

$$O_{P,i} = \Lambda_i \times (1 - H_{P,i}) \times Q_i, \ \forall\ i = 1, \ldots, K\ . \tag{1}$$

Table 1: Main notations for single cache and file f_i

Notation	Description
Λ_i	Arrival rate of file f_i
$1/\mu_i$	Expected value of TTL T_i of file f_i
$F_i(t)$	CDF inter-arrival times of file f_i
$G_i(t)$	CDF inter-miss times of file f_i
$T_i(t)$	CDF TTL duration of file f_i
$Z^*(s)$	LST of CDF $Z(t)$
$H_{P,i}, M_{P,i}$	Hit, miss probability resp. of file f_i
$H_{R,i}, M_{R,i}$	Hit, miss rate resp. of file f_i
$O_{P,i}$	Occupancy probability of file f_i

Moreover, if C is the cache of size we have

$$C = \sum_{i=1}^{K} \Lambda_i \times (1 - H_{P,i}) \times Q_i \ . \tag{2}$$

Proof Equation (1) follows directly by applying Little's law as follows. Since successive requests of file f_i are independent, the expected number of copies for file f_i in a cache is $O_{P,i}$, the rate at which file f_i enters the cache is the miss rate i.e. $\Lambda_i \times (1 - H_{P,i})$, and the expected time that file f_i spends in the cache is Q_i. Also, (2) follows from (1) by summing the two sides of the equality over all files and replacing $\sum_i O_{P,i}$ by the cache size. ∎

A similar result was established by Fricker et *al.* [11] for RND caches fed by Poisson request processes. Therefore, Lemma 1 extends results in [11] to all caches (e.g. LRU, FIFO, RND, etc.) fed by renewal request processes (i.e. under Assumption 1). The next result provides bounds of the characteristic time T_i, under the CTA i.e. when we approximate $E[T_i],\ \forall i$ by the same constant T.

Proposition 1 (Bounds of characteristic time) *When $E[T_i] \approx T,\ \forall i$, the following inequalities hold*

$$T \leq \quad E[Q_i] \quad , \ \forall\ i \tag{3}$$

$$\Lambda_i(1 - H_{P,i})T \leq \quad O_{P,i} \quad \leq \Lambda_i T, \ \forall\ i \tag{4}$$

$$T_{\min} = \frac{C}{\Lambda} \leq \quad T \quad \leq T_{\max} = \frac{C}{\Lambda \times \bar{m}_p} \tag{5}$$

where $\bar{m}_p = \sum_{i=1}^{K} \frac{\Lambda_i}{\Lambda}(1 - H_{P,i})$ denotes the average miss probability and $\Lambda = \sum_{i=1}^{K} \Lambda_i$ is the aggregate request rate.

Proof From the definitions of Q_i and T_i, one can easily see that $T_i \leq Q_i$. By taking the expectation on the latter inequality and then applying the CTA (i.e. $E[T_i] \approx T,\ \forall i$), we obtain (3). The first inequality in (4) follows directly by applying (3) and Eq.(1) of Lemma 1. By noting that $O_{P,i}$ is the expected number of occurrences of file i in the cache and $\Lambda_i T$ is the expected number of occurrences of file i that arrive on the cache within T, one can derive that $O_{P,i} \leq \Lambda_i T$. Taking the sum of (4) over all files and applying Eq.(2) of Lemma 1, the bounds of the characteristic time T in (5) are obtained with some basic algebra. ∎

Now we are ready to present our models for single cache approximation.

4 Single cache approximation

In this section, we develop CTA and PMC for FIFO, RND and LRU caches in isolation.

4.1 FIFO cache

Once modelled as a TTL-based cache (see Figure 2), the hit probability $H_{P,i}$ of file f_i in a FIFO cache may be approximated by $H_{P,i} = 1 - (1 + E[Z])^{-1}$ where $E[Z]$ is the expected number of hits during the sojourn time of the file in the cache (Cf. [21, Prop. 3]).

Since FIFO is a TTL-non-resetting policy, the application of the CTA yields that $E[Q_i] = E[T_i] \approx T$; and hence, $E[Z] = E\left[M_i(T_i)\right] \approx M_i(T)$. Therefore, it follows from Lemma 1 that

$$C = \sum_i O_{P,i} = \sum_i \lambda_i (1 + M_i(T_i))^{-1} Q_i \approx T \sum_i \lambda_i (1 + M_i(T))^{-1}$$

with T as an unknown. We propose an iterative procedure shown in Algorithm 1 to calculate T by solving the latter fixed-point equation. We initialize the iteration as follows. Since $Q_i = T$ it follows from Proposition 1 that

$$\frac{C}{\Lambda} \leq T = \frac{C}{\Lambda \bar{m}_p}.$$

Using the lower bound C/Λ as the initial value of T provides fast convergence. We observe from Figures 3(a) and 3(b), values of T are obtained after one iteration step.

Regarding the miss process of FIFO caches, the first two moments of the inter-miss times (needed for APA) are obtained from the LST $G_i^*(s)$ of the CDF $G_i(t)$ of inter-miss times Y_i provided in [21, Sect. 4.3.1].

$$E[Y_i] = E[X_i](1 + L_i^*(0)) \tag{6}$$

$$E[Y_i^2] = E[X_i^2](1 + L_i^*(0)) - 2E[X_i] \frac{\mathrm{d}L_i^*(s)}{\mathrm{d}s}\Big|_{s=0} \tag{7}$$

where $L_i^*(0) = L_i(\infty) = M_i(T)$, $L_i^*(s)$ and its derivative are given in Appendix B.

4.2 Random cache

The characteristic time of a RND cache is an exponentially distributed random variable which allows us to model a RND cache as either a TTL-renewing or TTL-non-renewing cache. By considering RND as a TTL-non-resetting policy, the hit probability can be approximated as $H_{P,i} = 1 - (1 + E[Z])^{-1}$ where $E[Z] = F_i^*(T_i^{-1})(1 - F_i^*(T_i^{-1}))^{-1} \approx F_i^*(T^{-1})(1 - F_i^*(T^{-1}))^{-1}$. Moreover, $E[Q_i] = E[T_i] \approx T$ and it follows from Lemma 1 that

$$C = \sum_i O_{P,i} = \sum_i \lambda_i (1 - F_i^*(T_i^{-1})) Q_i \approx \sum_i \lambda_i T (1 - F_i^*(T^{-1}))$$

with T as an unknown variable. Algorithm 1 implements a iterative scheme to compute the characteristic time T of RND cache with the initial point provided by Proposition 1

$$\frac{C}{\Lambda} \leq T = \frac{C}{\Lambda \bar{m}_p}.$$

As later confirmed by simulations (see Figures 3(a) and 3(b)), the characteristic times of RND and FIFO caches are approximately equal under identical traffic conditions. This suggests that we can safely replace FIFO caches by RND ones for the calculation of the characteristic time T;

Algorithm 1: Approximating characteristic time of a cache with FIFO or RND policy with a renewal process input

input : Cache Size C, cache policy P, number of files K, total arrival rate λ, precision ϵ
output: Characteristic time T

$n \leftarrow 1$
$T^{(1)} \leftarrow C/\lambda$
$\bar{m}_p^{(1)} \leftarrow 1$
do
  $n \leftarrow n+1$
  if $P =$ FIFO **then**
    $\bar{m}_p^{(n)} \leftarrow \sum_{i=1}^{K} \frac{\lambda_i}{\lambda}(1 + M_i(T^{(n-1)}))^{-1}$
  else
    $\bar{m}_p^{(n)} \leftarrow \sum_{i=1}^{K} \frac{\lambda_i}{\lambda}(1 - F_i^*(1/T^{(n-1)}))$
  $T^{(n)} \leftarrow C/(\lambda \times \bar{m}_p^{(n)})$
while $|T^{(n)} - T^{(n-1)}| > \epsilon$
$T \leftarrow T^{(n)}$

thereby avoiding the evaluation of the incomplete gamma function in the calculation of time of a FIFO cache (Cf. Appendix B).

Regarding the first two moments of the inter-miss times (needed for APA), modelling RND caches as TTL-resetting caches significantly simplifies their calculation since they are obtained deriving the LST $G_i^*(s)$ of inter-miss times Y_i given in [10, Sect. 3.2].

$$E[Y_i] = \frac{E[X_i]}{1 - F_i^*(\mu)} \tag{8}$$

$$E[Y_i^2] = \frac{E[X_i^2]}{1 - F_i^*(\mu)} - \frac{2 \times E[X_i]}{(1 - F_i^*(\mu))^2} \frac{\mathrm{d}L_i^*(s)}{\mathrm{d}s}\Big|_{s=0} \tag{9}$$

where $\mu = T^{-1}$; $L_i^*(s) = F_i^*(s+\mu)$, $L_i^*(0)$ and $\frac{\mathrm{d}L_i^*(s)}{\mathrm{d}s}\Big|_{s=0}$ are provided in Appendix C.

4.3 LRU cache

LRU is modeled as a TTL-renewing policy; therefore the hit $H_{P,i}$ probability of file f_i in a LRU cache can be approximated by $H_{P,i} = F_i(T_i) \approx F_i(T)$ thanks to the CTA. Applying Lemma 1 yields

$$C = \sum_i O_{P,i} = \sum_i \hat{F}_i(T_i) \approx \sum_i \hat{F}_i(T)$$

with T as unknown. The above equality is solved using the iterative scheme of Algorithm 2 by multiplying both members by T i.e.

$$T \approx \frac{C \times T}{\sum_i \hat{F}_i(T)},$$

and using the initial value C/Λ provided in the lower bound of Proposition 1. Similar to RND caches (see [10, Sect. 3.1]), the first and second moments of the inter-miss times Y_i are obtained

Algorithm 2: Approximating characteristic time of a cache with LRU policy with a renewal process input

input : CacheSize C, number of files K, total arrival rate λ, precision ϵ
output: Characteristic time T

$n \leftarrow 1$
$C^{(1)} \leftarrow C$
$T^{(1)} \leftarrow C^{(1)}/\lambda$
$\bar{m}_p^{(1)} \leftarrow 1$
do
    $n \leftarrow n+1$
    $C^{(n-1)} \leftarrow \sum_{i=1}^{K} \hat{F}_i(T^{(n-1)})$
    $T^{(n)} \leftarrow C \times T^{(n-1)}/C^{(n-1)}$
    $C^{(n)} \leftarrow \sum_{i=1}^{K} \hat{F}_i(T^{(n)})$
while $|T^{(n)} - T^{(n-1)}| > \epsilon$
$T \leftarrow T^{(n)}$

as following.

$$E[Y_i] = \frac{E[X_i]}{1-F_i(T)} \tag{10}$$

$$E[Y_i^2] = \frac{E[X_i^2]}{1-F_i(T)} - \frac{2 \times E[X_i]}{(1-F_i(T))^2} \frac{\mathrm{d}L_i^*(s)}{\mathrm{d}s}\Big|_{s=0} \tag{11}$$

where $L_i^*(s)$ and $\frac{\mathrm{d}L_i^*(s)}{\mathrm{d}s}\Big|_{s=0}$ are given in Appendix D.

4.4 Preliminary results and remarks

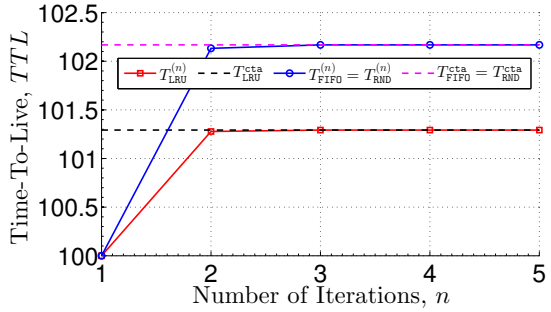
Before discussing the model for a network of caches, we validate Algorithms 1 and 2 on single FIFO/RND and LRU caches in isolation. We consider caches with capacity $C = 10^2$ and a catalog of $K = 10^5$ files fed by two types of request processes.

Poisson Process Here, requests are generated by a Poisson process with total request rate $\lambda = 1$, and assume that file popularity follows a Zipf distribution with $\alpha = 0.7$. Figure 3(a) show that our algorithms are accurate and fast. This stresses the importance of the initialization point to $T^{(0)} = C/\Lambda$ which makes our algorithms converge after one iteration.

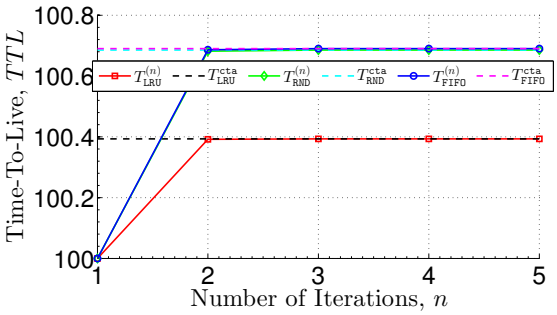
Interrupted Poisson Process In this case, traffic is described by Interrupted Poisson Processes (IPP), which is a renewal process having hyper-exponentially distributed inter-arrival times. The total request rate is set to $\lambda = 1$, and the content popularity follows a Zipf distribution with $\alpha = 0.7$. The squared coefficient of variation of each IPP is chosen to be $c_v^2 = 1.5$. Figure 3(b) shows that our algorithms converge very fast to the real value of the characteristic time.

We conclude this section with two remarks.

Remark 1 ($T_{\texttt{LRU}} \neq T_{\texttt{FIFO}}$) Unlike what was stated in [19], the characteristic time of a FIFO cache, $T_{\texttt{FIFO}}$, is not in general equal to that of a LRU cache, $T_{\texttt{LRU}}$; but, $T_{\texttt{LRU}} \leq T_{\texttt{FIFO}}$. This is



(a) IRM traffic/Poisson processes



(b) Renewal traffic/Interrupted Poisson Processes

Figure 3: Approximation of the Characteristic times of FIFO, RND and LRU caches.

because $T_{\texttt{LRU}}$ measures the time required to observe C distinct requests, while $T_{\texttt{FIFO}}$ is the time to observe C **cache misses**. For a FIFO cache in the steady state, a cache hit will not change the state of the cache. This means that files are not pushed down when a cache hit occurs (as the case for LRU caches) and hence the characteristic time of a FIFO cache is at least equal to that of a LRU cache with same size C, i.e. $T_{\texttt{LRU}} \leq T_{\texttt{FIFO}}$. Figure 3(a) confirms this result.

Remark 2 (Poisson assumption) When the arrival request for file i is assumed to follow a Poisson process of rate λ_i, the occupancy probability $O_{P,i}$ equals the hit probability $H_{P,i}$. In this case, Q_i and T_i are related through Lemma 1 as follows

$$Q_i = \lambda_i^{-1} \times \frac{H_{P,i}}{1 - H_{P,i}}, \ \forall \ i \tag{12}$$

where $H_{P,i}$ is a function of T_i (or T by applying the CTA i.e. $E[T_i] = T, \forall \ i$) given by

$$H_{P,i} \approx \begin{cases} 1 - (1 + \lambda_i T)^{-1} & \text{, for FIFO caches} \\ 1 - (1 + \lambda_i \mu^{-1})^{-1} & \text{, for RND caches} \\ 1 - e^{-\lambda_i T} & \text{, for LRU caches} \end{cases} \tag{13}$$

Finally, we obtain the characteristic time equations under Poisson request processes which have been solved using Taylor's expansion [16], inverse function [4], and more generally in this paper

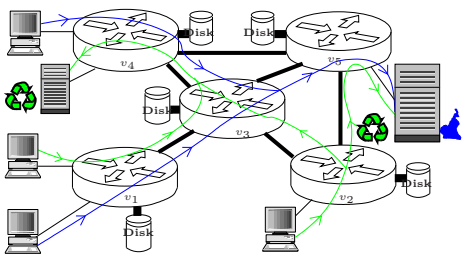


Figure 4: Request flows merge, split, move in opposite directions at some nodes.

through a fixed-point iterative method.

$$C = \sum_i H_{P,i} \approx \begin{cases} \sum_i 1 - (1+\lambda_i T)^{-1} & \text{, for FIFO caches} \\ \sum_i 1 - (1+\lambda_i \mu^{-1})^{-1} & \text{, for RND caches} \\ \sum_i 1 - e^{-\lambda_i T} & \text{, for LRU caches} \end{cases} \tag{14}$$

Unfortunately, the applicability of Remark 2 on cache networks is limited since miss streams are no longer described by Poisson processes.

5 Network approximation

In this section, we extend our single cache approximation to a heterogeneous cache network with arbitrary topology. First, we address the case where the network topology is a Polytree; then we analyse arbitrary cache network. Recall that each file is permanently stored in at least one server connected to the network and requests of each file are routed as a feed-forward network as shown in Figure 4.

In the network of Figure 4, we have two content items: file f_1 is green and file f_2 is blue. The blue content item is located at the server connected to node v_5; requests for it are routed on the tree formed by nodes v_1 and v_4 (the leaf caches), v_3 (the intermediate cache), and v_5 (the root cache). Meanwhile the green content item is available at two different servers and its requests are routed on the polytree having nodes v_4 and v_5 as roots. We will in general refer to this tree or polytree based routing as the *routing topology*. Each node can independently choose to implement LRU, FIFO or RND for managing the cache content.

Finally, we assume that Assumption 1 holds i.e. exogenous and aggregated request streams may be approximated by renewal processes.

5.1 Modelling network primitives

The main challenges when extending the analysis of single caches to cache networks are to describe the following network primitives in terms of operations on processes.

5.1.1 Splitting a stream of requests

When leaving a cache (e.g. at node v_2 in Figure 4), missed requests may be routed towards one or many destinations. This network primitive can be modelled as a *dependent thinning of request point processes* [13]. Let $\mathcal{N}(i)$ denote the set of neighbour caches fed by missed requests at cache i. In our toy example, cache i is node v_2 and $\mathcal{N}(2) = \{v_3, v_5\}$. We denote by $p_{i,j}(k)$ the probability that cache i forwards a missed request of file $f_k, k = 1, 2$ to cache $j \in \mathcal{N}(i)$ and

we assume $\sum_{j\in\mathcal{N}(i)} p_{i,j}(k) = 1\ ,\ \forall(i,k)$. The two first moments of inter-request times $Y_{i,j,k,\texttt{th}}$ in the thinned request process of file f_k that arrives at cache j from cache i are characterized from the original miss process of file f_k at cache i as follows.

$$\begin{aligned} E[Y_{i,j,k,\texttt{th}}] &= \frac{E[Y_{i,k}]}{p_{i,j}(k)} \\ E[Y^2_{i,j,k,\texttt{th}}] &= \frac{E[Y^2_{i,k}]}{p_{i,j}(k)} + 2\frac{1-p_{i,j}(k)}{p_{i,j}(k)^2}(E[Y_{i,k}])^2 \end{aligned}$$

5.1.2 Requests flowing in "opposite" direction

This situation occurs between nodes v_3 and v_4 in Figure 4 where miss streams of cache v_3 are routed towards cache v_4 and vice versa. Let us denote by T_j and C_j the characteristic time and the storage capacity at cache $j = v_3, v_4$, $F_{0,j,k}(t)$ and $\Lambda_{j,k}$ the CDF of inter-arrival times and the rate of the overall requests for file $f_k, k = 1, 2$ at cache j respectively, and $G_{i,j,k}(t)$ the CDF of inter-miss times of requests for file f_k at cache i forwarded to cache j. Given that $G_{i,j,k}(t)$ is a function of $F_{0,i,k}(t)$ and T_i (see [10, Sect. 3.1 & Sect. 3.2] for LRU/RND caches, and [21, Prop. 2] for FIFO caches), then applying Lemma 1 and invoking the CTA at cache $j = v_4$ (and we set $i = v_3$), we get

$$C_j \approx \begin{cases} \frac{\Lambda_{j,2}T_j}{1+M_{0,j,2}(T_j)} + \frac{\Lambda_{j,1}(1+M_{0,i,1}(T_i))^{-1}T_j}{1+M_{i,j,1}(T_j)} & \text{, if FIFO} \\ \hat{F}_{0,j,2}(T_j) + \hat{G}_{i,j,1}(T_j) & \text{, if LRU} \\ \frac{1-F^*_{0,j,2}(T_j^{-1})}{1/(\Lambda_{j,2}T_j)} + \frac{1-G^*_{i,j,1}(T_j^{-1})}{1/(\Lambda_{i,1}(1-F^*_{0,i,1}(T_i^{-1}))T_j)} & \text{, if RND} \end{cases} \tag{15}$$

A similar set of equations can be derived if caches v_3 and v_4 have different replacement policies. Hence, the characteristic times at these nodes are dependent/coupled; and therefore, T_3 and T_4 may not be approximated by just running Algorithms 1 or 2 in one shot as we did in Section 4 for caches in isolation. However, by combining equations (15) for caches v_3 and v_4, we obtain a coupled system of fixed-point equations that can be solved simultaneously using an iterative scheme (Cf. Algorithm 4).

5.1.3 Merging streams of requests

This network primitive is illustrated in Figure 4 at nodes v_3 and v_5 where independent (we assume) request streams (that consist of miss streams of other caches and, if any, exogenous requests) are superimposed. We rely on the APA to describe each aggregate request process as a simple renewal process (see Section 3) based on the two first moments of inter-request times.

Whitt proposed two techniques, namely the stationary interval method and the asymptotic method to approximate the superposition of independent renewal processes [27, Sect. 4]. The former calculates the exact two first moments of the first inter-arrival time of the aggregated process using its exact CDF (see [18, Eq. 4.1]); while the latter method uses the asymptotic results of the two first moments of the time-average renewal rate over a finite interval of time (see [25, Thm. 5.1 and 5.2]).

Qualitatively speaking, both methods return the identical aggregated request rate; however, they differ on the calculation of the squared coefficient of variation (scv), denoted by c_v^2, of inter-request times in the aggregated process. The stationary interval method requires that each request stream to be aggregated be approximated by a hyper/shifted exponential renewal process. Then it takes advantage of this simple structure to approximate c_v^2 by a constant denoted c_s^2 and calculated with the exact CDF of the first inter-request time [18]. On the other hand the

asymptotic method approximates c_v^2 by a constant c_a^2 defined as a weighted sum of scv of request stream components being superimposed, where the weight is the ratio of the component request rate and the aggregated rate.

Hybrid method and weighting function In this paper, we introduce a Hybrid method (similar to that described by Albin for Queueing systems [2]). Let $c_p^2 = 1$ denote the scv of the inter-request times distribution of the aggregated streams if it is approximated by a Poisson process. We approximate c_v^2 by a constant c_h^2 calculated as a convex combination of c_a^2 (resp. c_s^2) and c_p^2.

More formally, if we consider a cache with in-degree d (including the exogenous process) and file f_k is requested (from this cache) by its $j^{\text{th}} = 1, \ldots, d$ neighbour, or exogenous process if any, with rate $\lambda_{j,k}$ then

$$c_h^2 = w_k c_a^2 + (1 - w_k) c_p^2, \quad (\text{resp. } = w_k c_s^2 + (1 - w_k) c_p^2) \tag{16}$$

where w_k is the weighting function related to file f_k. This function is calculated as follows

$$w_k = \frac{1}{1 + (1 - O_{P,k})^2 \times d_{\text{eff}}}, \quad O_{P,k} \approx \Lambda_k \frac{C}{\Lambda}, \tag{17}$$

with $\Lambda_k = \sum_{j=1}^{d} \lambda_{j,k}$ and $O_{P,k}$ are the total request rate and the occupancy probability of file f_k in the cache, $\Lambda = \sum_{k=1}^{K} \Lambda_k$ and C are the total request rate and the cache capacity, and d_{eff} is the **effective degree** of our cache [2, Eq. (2)] given by

$$d_{\text{eff}} = \left[\sum_{j=1}^{d} \left(\frac{\lambda_{j,k}}{\Lambda_k} \right)^2 \right]^{-1} \tag{18}$$

We can easily check that if $O_{P,k} \to 1$ and d_{eff} fixed, then the weighting function $w_k \to 1$ and the Hybrid method reduces to Whitt's asymptotic method. This case occurs mainly for popular files. Meanwhile, as $d, d_{\text{eff}} \to \infty$ the weighting function $w_k \to 0$ and the Hybrid method converges to the Poisson approximation.

5.2 Cache network with polytree topology

In this section, we consider a polytree-based cache network, where servers are located at root nodes and requests flow in the same direction from leaf nodes to the roots.

Under this settings, we can analyze our network by starting from the leaf nodes and approximating the characteristic time and the miss processes. The latter processes are then aggregated with the exogenous request streams if any to form the arrival process at a higher level cache. We repeat this process until we reach all root nodes. The algorithm for approximating performance of cache networks with polytree topology is given in Algorithm 3.

Before discussing the algorithms for the general network, we make the following remarks regarding our routing schemes.

Remark 3 (Deterministic Routing) General cache networks studied in [23] consider at least one server in the network for each content, one path from any cache to any server, and routing is done over shortest paths. In the case of multiple shortest paths, one is chosen uniformly at random. This scheme is a special case of our probabilistic routing on per-file feed-forward networks. In fact, if $\mathcal{N}_{\mathtt{s}}(i,k)$ is the set of different caches which correspond to the next hop on one of the shortest paths from cache i to a server of file f_k, it is enough to take $p_{i,j}(k) = 0$, $\forall j \in \mathcal{N}(i) \backslash \mathcal{N}_{\mathtt{s}}(i,k)$ and $p_{i,j}(k) = 1/|\mathcal{N}_{\mathtt{s}}(i,k)|$, $\forall j \in \mathcal{N}_{\mathtt{s}}(i,k)$.

Algorithm 3: Approximating performance of a cache network with polytree topology

input : Topology $\mathcal{G}(V, E)$, Depth d, Number of files K, Exo. processes $\{F^*_{0,i,k}(s),\ v_i \in V, f_k \in \mathcal{F}\}$, Size of caches $\{C_i, v_i \in V\}$

output: Characteristic times T_i, hit probs $H_{P,i,k}$, occupancy $O_{P,i,k}$, $\{v_i \in V, f_k \in \mathcal{F}\}$

while $d \neq 0$ **do** ; // d = 0 for server

  $\mathcal{S}(d) \xleftarrow{\texttt{Select caches at depth } d} \mathcal{G}(V, E)$

  foreach $i \in \mathcal{S}(d)$ **do** ; // Start from leaves

    for $k = 1, \ldots,\ K$ **do** ; // On each file

      $\{\Lambda_{i,k}, H^*_{.,i,k}(s)\} \xleftarrow{\texttt{APA}} \{F^*_{0,i,k}(s), \nu_{j,i,k}, G^*_{j,i,k}(s), j \in \mathcal{C}(i)\}$;

    end

    $\{T_i\} \xleftarrow{\texttt{Algs. 1 or 2}} \{C_i,\ H^*_{.,i,k}(s),\ k = 1, \ldots, K\}$;

    for $k = 1, \ldots,\ K$ **do** ; // On each file

      $\{H_{P,i,k}, O_{P,i,k}\} \xleftarrow{\texttt{PMC}} \{T_i, H^*_{.,i,k}(s)\}$;

      $G^*_{i,.,k}(s) \xleftarrow{\texttt{PMC}} \{T_i, H^*_{.,i,k}(s)\}$;

      $\{G^*_{i,j,k}(s),\ j \in \mathcal{N}(i)\} \xleftarrow{\texttt{RRS}} G^*_{i,.,k}(s)$;

      $\{G_{i,j,k}(t), G^*_{i,j,k}(s)\} \xleftarrow{\texttt{APA}} G^*_{i,j,k}(s)$;

    end

  end

  $d \leftarrow d - 1$;

end

5.3 Cache network with arbitrary topology

Thanks to our toy network in Figure 4, we have clearly shown that calculating the performance of general cache networks requires solving a coupled system of fixed-point equations, especially when requests flow in opposite directions (Cf. Section 5.1.2).

An intuitive approach would be to derive these coupled equations for each cache network and then solve them through an iterative method. However, this approach will be a per-case solution. Instead, we develop an iterative algorithm that finds the characteristic times without deriving the coupled system of equations (15) for general and heterogeneous cache network.

Our cache network approximation algorithm starts with an initialization step where all caches are assumed to have miss probabilities of one. The consequence is that the miss process of a node is initialized by its aggregated arrival processes. Then the TTLs at each cache are also initialized to the minimum TTL values provided by Proposition 1 i.e. the ratio of the cache capacity and the total request rate on the cache. After, this initialization step, our algorithm updates the miss processes of the caches using the initial value of the TTLs; and recalculates the new TTL values as described in Algorithms 1 and 2. Our cache network algorithm halts when all TTL values at all nodes of the network have converged. In this way we can iteralively solve any system of coupled equations and handle arbitrary cache network. This approach is presented in Algorithm 4.

Algorithm 4: Approximation on arbitrary physical network of caches with precision $\epsilon > 0$

input : `CacheSize` C_i, `CatalogSize` K, `ExoRates` $\lambda_{i,k}$
output: `CharacteristicTime` T_i

$n := 1$;
for $i = 1, \ldots, N$ **do** ; `// On each cache`

  for $k = 1, \ldots, K$ **do** ; `// On each file`

    $m^{(1)}_{p,i,k} := 1$;
    $\Lambda^{(1)}_{i,k} := \lambda_{i,k} + \sum_{v_j \,:\, v_i \in \mathcal{N}(j)} \lambda_{j,k}$;
    $H^{*(1)}_{.,i,k}(s) \xleftarrow{\texttt{APA}} \{F^*_{0,i,k}(s),\ F^*_{0,j,k}(s),\ v_j : v_i \in \mathcal{N}(j)\}$;
    $T^{(1)}_i := C_i / (\sum_k \Lambda_{i,k})$;
  end
end
while `true` **do** ; `// Iterate until convergence`

  $n := n + 1$;
  for $i = 1, \ldots, N$ **do** ; `// On each cache`

    for $k = 1, \ldots, K$ **do** ; `// On each file`

      $\{G^{*(1)}_{i,j,k}(s),\ v_j \in \mathcal{N}(i)\} \xleftarrow{\texttt{PMC \& RRS}} \{H^{*(1)}_{.,i,k}(s), T^{(1)}_i\}$;
      $H^{*(n)}_{.,i,k}(s) \xleftarrow{\texttt{APA}} \{F^*_{0,i,k}(s),\ G^{*(n-1)}_{j,i,k}(s),\ v_j : v_i \in \mathcal{N}(j)\}$;
      $\{H^{(n)}_{P,i,k}, O^{(n)}_{P,i,k}, m^{(n)}_{p,i,k}, \Lambda^{(n)}_{i,k}\} \xleftarrow{\texttt{PMC}} \{H^{*(n)}_{.,i,k}(s), T^{(n-1)}_i\}$;
    end
    $m^{(n)}_{R,i} := \sum_{k=1}^{K} \Lambda^{(n)}_{i,k} m^{(n)}_{p,i,k}$;
    $C^{(n)}_i := \sum_{k=1}^{K} O^{(n)}_{P,i,k}$;
    if `RND or FIFO` **then**
      $T^{(n)}_i := \frac{C_i}{m^{(n)}_{R,i}}$;
    else
      $T^{(n)}_i := \frac{C}{C^{(n)}_i} \times T^{(n-1)}_i$;
    end
    if $\frac{1}{N} \sum_{i=1}^{N} (T^{(n)}_i - T^{(n-1)}_i)^2 < \epsilon$ **then**
      $T_i :\approx T^{(n)}_i$;
      `break`;
    end
  end
end

6 Evaluation results

In this section, we evaluate the accuracy of our model by comparing its predictions for the per-file hit probabilities and the average miss probability at each cache to those obtained via event-driven simulations (≈ 16.77 million of events generated).

If the per-file hit probabilities are interesting from the user point of view, the per-cache average miss probabilities are performance metrics of interest of the system point. In fact, the miss probability provides insights on the residual load placed on end-servers. In order to appreciate the quality of our model, we also calculate the miss probability ratio (MPR) between simulation results and our predictions. A miss probability ratio close to one is desirable and will be a global indicator of the quality of our model.

We assume that exogenous requests arrive at a nodes of the network according to a Poisson process with rate $\lambda = 1$, and the file popularity follows a Zipf distribution with parameter $\alpha = 0.7$.

6.1 Poisson approximation and a-NET model

In this section, we assume that request streams are described by Poisson processes. This workload model is commonly used in the literature [23, 19]. The Poisson approximation is obtained from our model by setting the weighting functions to zero ($w_k = 0, \forall k = 1, \ldots, K$). We evaluate our Poisson approximation and a-NET model on random network and binary tree of LRU caches.

In the random network case, links are drawn uniformly at random as shown in Figure 5(c) and we have exogenous Poisson requests arriving at all caches. For sake of presentation we only report the file hit probabilities at the node directly connected to the servers and also the Miss Probability Ratio (MPR) at each cache.

First, we observe that our Poisson approximation and the a-NET model [23] predict almost identical per-file hit probabilities and differ slightly in their estimates of the MPR as illustrated in Figure 5. In fact, the single cache approximation [8] used in the a-NET model is as accurate as the CTA implemented in our model. Therefore, it is not surprising that they produce similar results under the same Poisson assumption. Second, the Poisson approximation and a-NET predictions are really close to the simulation results as we can see a good match on hit probabilities and that the MPR is close to one.

We observe in other experiments that for caches with large in-degree, the discrepancy between the Poisson approximation (or the a-NET model) and the simulation results decreases as the in-degree of the cache increases. This observation is consistent with the fact that the aggregation of many point processes converges to Poisson process as the number of point processes increases.

For the binary tree, we assume that requests are generated at leaves only. We report the metrics of interest at the root (i.e. node 1) where the server is attached.

Figure 6 shows that the Poisson approximation performs poorly and tends to overestimate the performance. This result contrasts with the ones observed on the previous experiment. It can be explained by the presence of exogenous Poisson streams at all caches of the random network where Poisson arrivals were the major contributor (thanks to its memoryless property and rate) to the aggregated request process of each cache. Therefore, Poisson approximation and a-NET are quite inaccurate on caches fed by non-Poisson processes.

6.2 Accuracy of Whitt approximations

In this section, we assume that request streams are described as in Section 3. Moreover, we set the weighting functions to one ($w_k = 1, \forall k = 1, \ldots, K$); hence, our Hybrid method reduces to

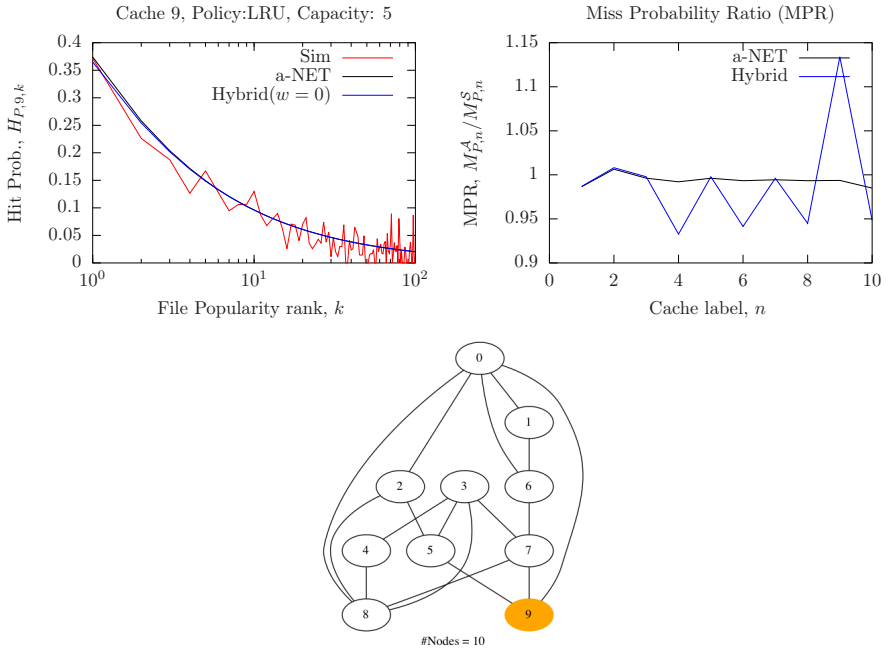


Figure 5: Simulations, Poisson approximation, and a-NET on 10 caches random network. Servers connected at node 9. Number of files $K = 10^2$, Cache capacity $C = 5$.

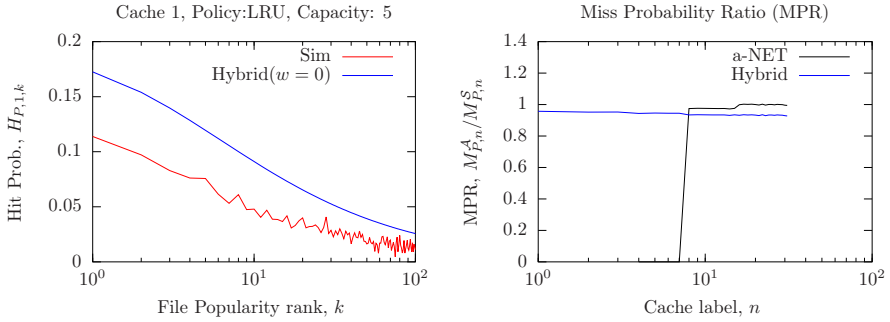


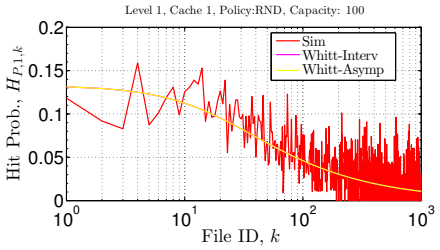
Figure 6: Poisson approximation on binary tree of depth 3 with 31 LRU caches. Servers connected at node 1. Number of files $K = 10^2$, Cache capacity $C = 5$.

Whitt's method (Cf. Eq. (16)). We refer to *Whitt-Interv* (resp. *Whitt-Asymp*) approximation our Hybrid model when relying on Whitt's stationary interval (resp. asymptotic) method given that $w_k = 1, \forall k$.

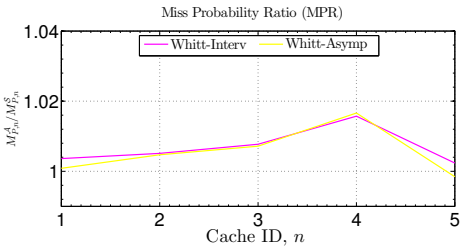
Linear networks We consider a tandem of caches where exogenous requests occur at the leaf node only. In this case, the overall request process at each cache is not a superposition of renewal processes but a simple renewal process. Our interest in this case study is to evaluate the accuracy of renewal approximation itself (i.e. without the aggregating effect) which is based on

the description of request streams by hyper/shifted-exponential renewal processes.

First, we assume that all nodes run the RND replacement policy. As expected, we can easily check that the asymptotic and the stationary interval methods predict identical performance metrics. Figure 8 shows that the approximate metrics of interest are quite good and close to the simulation results. We also observed such accuracy on a tandem network of five FIFO caches. This means that the shifted-exponential renewal processes introduced in Section 3 are appropriate models rather than Poisson processes to describe the miss streams of RND and FIFO caches.



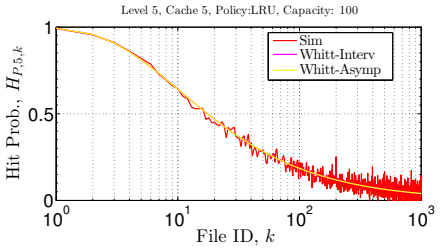
(a) Cache 1, root node



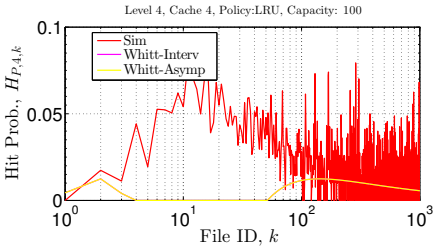
(b) Miss Probability Ratio

Figure 7: Whitt approximations on a linear network of five RND caches, $K = 10^3$, $C = 10^2$.

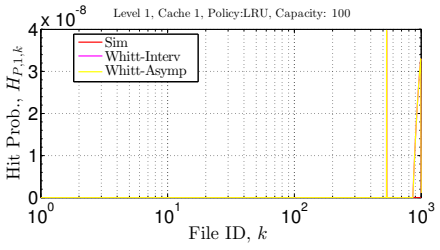
Second, we simulated a tandem network of five LRU caches where requests arrive at the leaf. We focus on the leaf node and the one after (i.e. fed by the leaf node) since simulations showed that metrics of interest are almost zero at the other nodes.



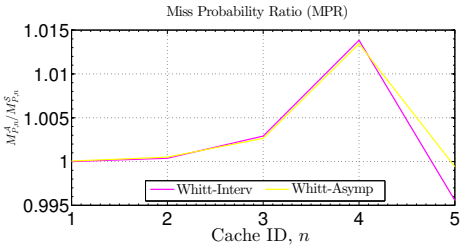
(a) Cache 5, leaf node



(b) Cache 4, parent of the leaf



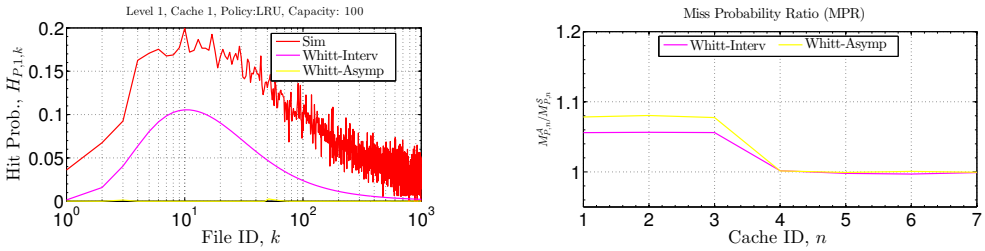
(c) Cache 1, root node



(d) Miss Probability Ratio

Figure 8: Whitt approximations on a linear network of five LRU caches, $K = 10^3$, $C = 10^2$.

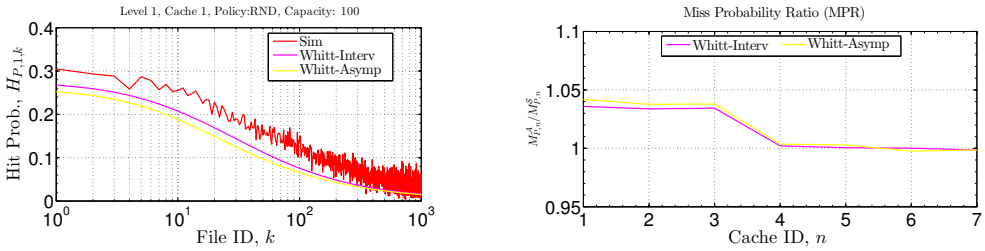
We observed significant discrepancy on per-file hit probabilities at cache 4 (parent of the leaf) between Whitt approximations and the simulation results.



(a) Cache 1, root node (b) Miss Probability Ratio

Figure 9: Whitt approximations on a binary tree of LRU caches, Depth 3, $K = 10^3$, $C = 10^2$.

Binary trees We consider a binary tree of depth three having seven LRU caches where requests occur at leaf nodes only. The goal of this experiment is to evaluate the quality of the Whitt stationary-interval and asymptotic methods in the presence of aggregated request streams. We report the per-file hit probabilities at the root and the per-cache MPRs. We observe in Figure 9 that *Whitt-Interv* and *Whitt-Asymp*) approximations perform poorly and underestimate the metrics of interest. The same observation was done in Figure 10 where caches are running the RND replacement policy.



(a) Cache 1, root node (b) Miss Probability Ratio

Figure 10: Whitt approximations on a binary tree of RND caches, Depth 3, $N = 10^3$, $C = 10^2$.

General networks We simulate the random network in Figure 5(c) with Poisson requests occurring exogenously at all caches (resp. at caches 2, 3, 5, and 7). We compare each of Whitt approximations against simulation results and a-NET/Poisson approximation.

Figure 11 shows that the *Whitt-Interv* approximation provides good estimates of performance metrics when requests occurs at edge nodes (i.e. caches 2, 3, 5, and 7) only. However, it performs poorly when assuming exogenous Poisson request streams at each cache as shown in Figure 12.

In both experiments, we observe that *Whitt-Asymp* approximation is more accurate than others as shown in Figures 13 and 14 respectively.

However, we recall that the *Whitt-Asymp* approximation provides poor predictions on linear and tree networks of LRU caches (as previously observed in Figure 9); thus, *Whitt-Interv* approximation would be preferable to analyze trees of LRU caches where requests occur on leaves. These observations help us to understand that the main source of error predictions is the approximation of the scv of aggregate processes for handling all network configurations. It is the purpose of the Hybrid approximation we described in Section 5.1.3.

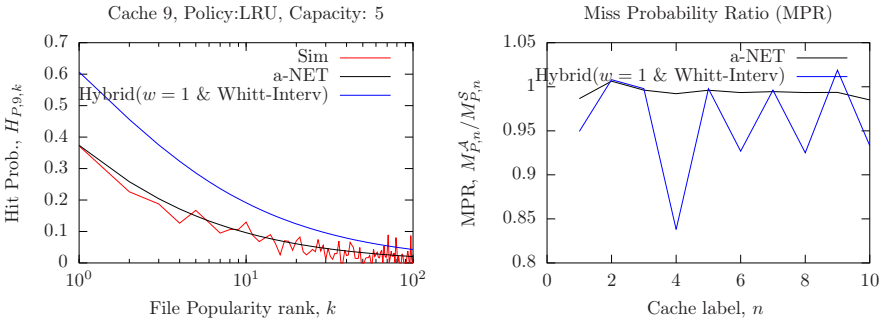


Figure 11: Simulations, a-NET/Poisson, and *Whitt-Interv* approximations on the ten caches random network **with exogenous requests at all caches**. $K = 10^2$ and $C = 5$.

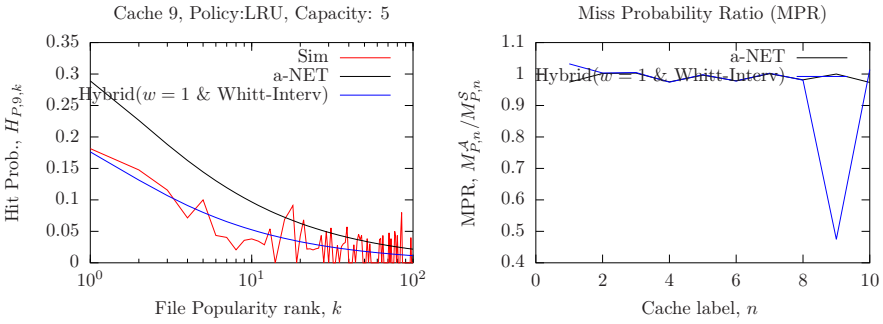


Figure 12: Simulations, a-NET/Poisson, *Whitt-Interv* approximations on the ten caches random network **with exogenous requests at caches 2, 3, 5, and 7**. $K = 10^2$ and $C = 5$.

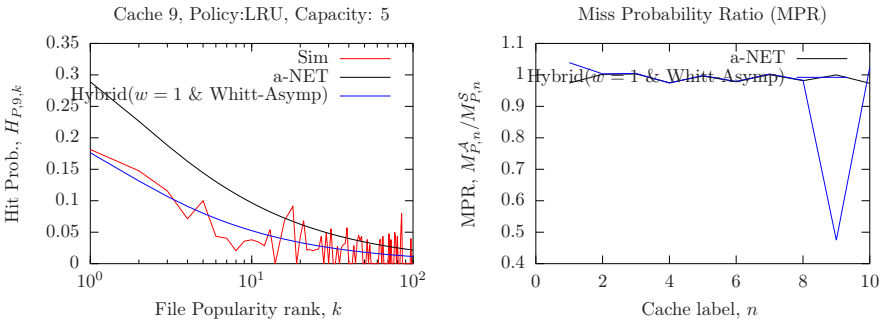


Figure 13: Simulations, a-NET/Poisson, *Whitt-Asymp* approximations on the ten caches random network **with exogenous requests at caches 2, 3, 5, and 7**. $K = 10^2$ and $C = 5$.

6.3 Accuracy of the Hybrid approximations

In this section, we evaluate the Hybrid approximation introduced in Section 5.1.3. First, we simulate the case where our Hybrid approximation is implemented with the stationary interval

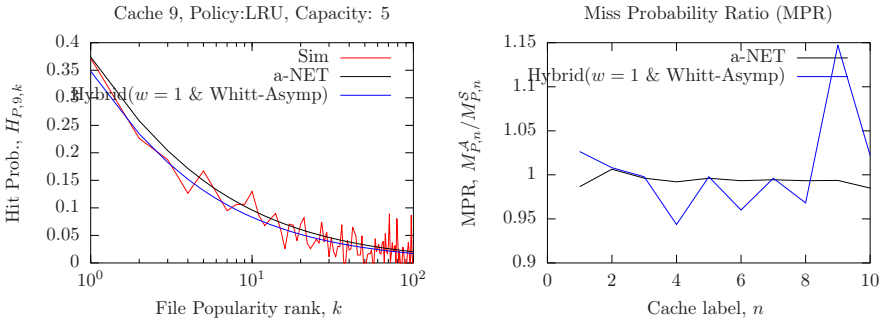
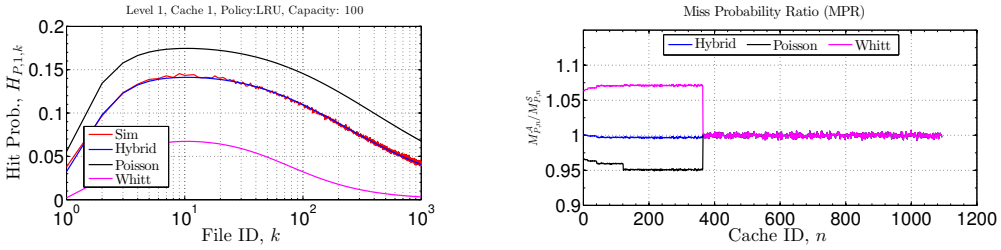


Figure 14: Simulations, a-NET/Poisson, *Whitt-Asymp* approximations on the ten caches random network **with exogenous requests at all caches**. $K = 10^2$ and $C = 5$.



(a) Cache 1, root node

(b) Miss Probability Ratio

Figure 15: Comparison of simulations, Poisson, *Whitt-Interv*, and *Hybrid-Interv* approximations on a ternary tree of 1093 LRU caches, Depth 7, $N = 10^3$, $C = 10^2$.

method (this approximation will be referred to as the *Hybrid-Interv* approximation). And second, we report results when the asymptotic method is used in the Hybrid approximation (this approximation will be referred to as *Hybrid-Asymp* approximation).

We observe that the *Hybrid-Asymp* approximation is less sensitive to network configurations and more accurate than the a-NET/Poisson, Whitt and *Hybrid-Interv* approximations on all networks we presented in this paper.

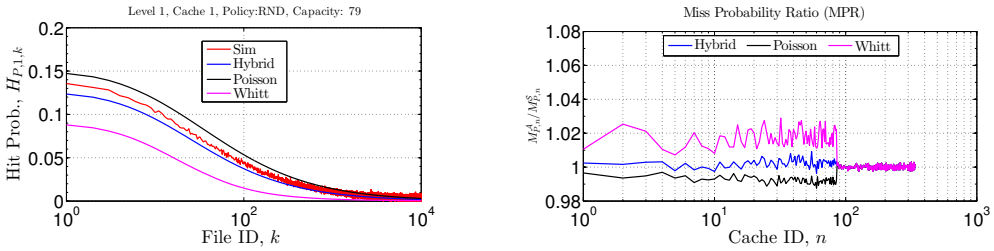
6.3.1 Using the stationary interval method: *Hybrid-Interv* approximation

For this case, we recall that the scv of inter-request times c_h^2 is computed as a *convex* combination of c_s^2 obtained from the *stationary-interval* method and c_p^2 ($c_p^2 = 1$): $c_h^2 = wc_s^2 + (1-w)c_p^2$, where $w \in [0,1]$ is given in (17).

Homogeneous tree: LRU policy everywhere We consider a ternary tree network of depth seven having 1093 LRU caches where requests occur on leaf nodes only. We compare Poisson, *Whitt-Interv*, and *Hybrid-Interv* predictions to the simulation results. As we can see in Figure 15, the Hybrid approximation performs the best and predicts all performance metrics with high accuracy.

In all of the homogeneous k-ary tree networks we tested, our Hybrid approximation accurately predicts all performance metrics with good quality indicators i.e. the curve of the per-cache miss probability ratio close to one.

Heterogeneous trees: mix of replacement policies, variable cache capacity, and larger number of files We consider a 4-ary tree of depth five having 341 caches. The cache capacities and the replacement policies are respectively chosen uniformly at random within the interval $[50; 150]$ and among the FIFO, RND, and LRU policies. The catalog size is set to $N = 10^4$.



(a) Cache 1, root node

(b) Miss Probability Ratio

Figure 16: Comparison of simulations, Poisson, *Whitt-Interv*, and *Hybrid-Interv* approximations on a heterogeneous ternary tree of 341 caches, Depth 5, $N = 10^4$, $C \in [50; 150]$.

We observe in Figure 16 that the *Hybrid-Interv* is quite accurate for predictions of the file hit probabilities and the per-cache MPRs on this heterogeneous tree network.

General networks: LRU policy at all nodes We run experiments on the random cache network in Figure 5(c) where: (i) request arrive at caches 2, 3, 5, and 7; then (ii) requests arrive at all caches. We report results at cache 9 where servers are attached.

In the first simulation, we observed that the *Hybrid-Interv* approximation is better that the a-NET/Poisson approximation (see Figure 17) and provides a similar level of accuracy to *Whitt-Interv* approximation (previously shown in Figure 12).

However in the presence of exogenous Poisson streams, the second experiment reveals that *Hybrid-Interv* approximation does not improve that much the *Whitt-Interv* approximation (previously shown in Figure 11) and remains less accurate than the a-NET/Poisson approximation as shown in Figure 18.

Therefore, the *Hybrid-Interv* approximation is not robust enough to handle all network configurations.

6.3.2 Using the asymptotic method: *Hybrid-Asymp* approximation

We repeated the experiments on the previous large tree of 1093 LRU caches, heterogeneous tree, and random cache networks. We observe a similar level of accuracy on these trees when relying on the *Hybrid-Asymp* approximation as shown in Figures 15 and 16 for the case where the *Hybrid-Interv* approximation is used. Hence, we report only the simulations on the random network where *Hybrid-Interv* approximation was inaccurate.

Figure 19 shows that the *Hybrid-Asymp* approximation is more accurate than the a-NET/Poisson approximation and predicts good estimates of the file hit probabilities on the random network with request arrivals at edge nodes only. Moreover, its accuracy is not affected by the presence of exogenous Poisson request streams at all caches as shown in Figure 20.

Finally, we assess the quality of the *Hybrid-Asymp* approximation on a large random network of $N = 10^2$ LRU caches. We set the cache capacity to $C = 10$ and the catalog of size to $K = 10^3$. We also assume exogenous Poisson request streams at all caches. Figure 21 shows the *Hybrid-Asymp* approximation provides better estimates of the file hit probabilities with per-cache MPR around one.

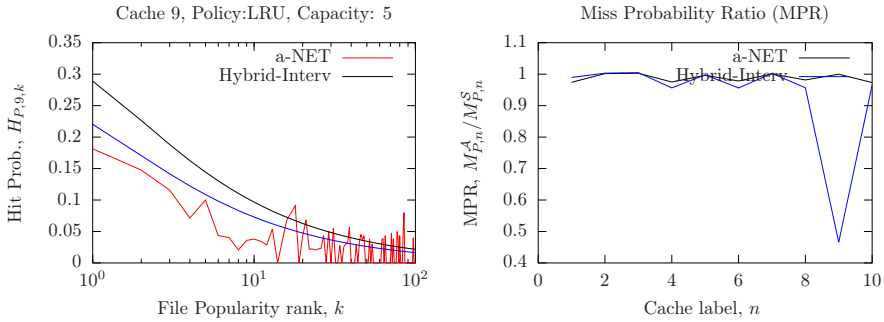


Figure 17: Simulations, a-NET/Poisson, and *Hybrid-Interv* approximations on the ten caches random network **with exogenous requests at caches 2, 3, 5, and 7**. $K = 10^2$ and $C = 5$.

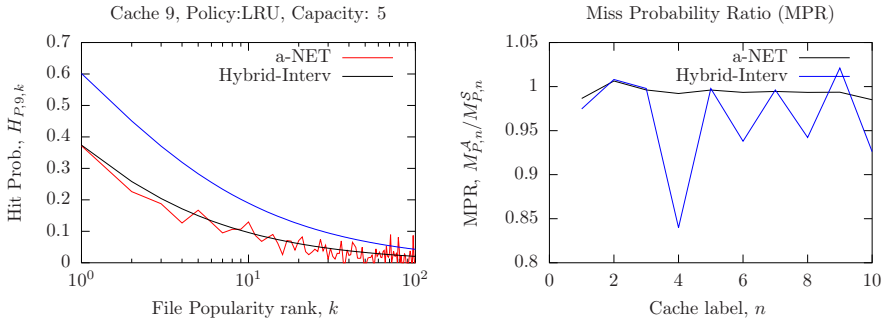


Figure 18: Simulations, a-NET/Poisson, and *Hybrid-Interv* approximations on the ten caches random network **with exogenous requests at all caches**. $K = 10^2$ and $C = 5$.

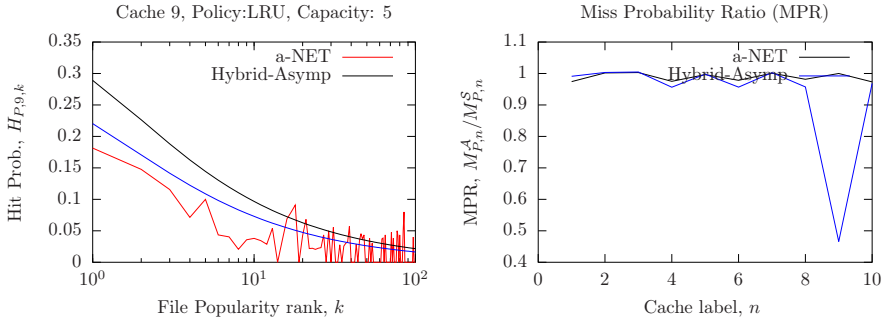


Figure 19: Simulations, a-NET/Poisson, and *Hybrid-Asymp* approximations on the ten caches random network **with exogenous requests at cache 2, 3, 5, and 7**. $K = 10^2$ and $C = 5$.

The *Hybrid-Asymp* approximation appears to be less sensitive to the network settings than others and provide a fairly good level of accuracy in all scenarios we tested. Our evaluation results establish that *Hybrid-Asymp* approximation is "superior" in terms of accuracy and versatility over the a-NET/Poisson, *Whitt-Interv*, *Whitt-Asymp*, and *Hybrid-Interv* approximations.

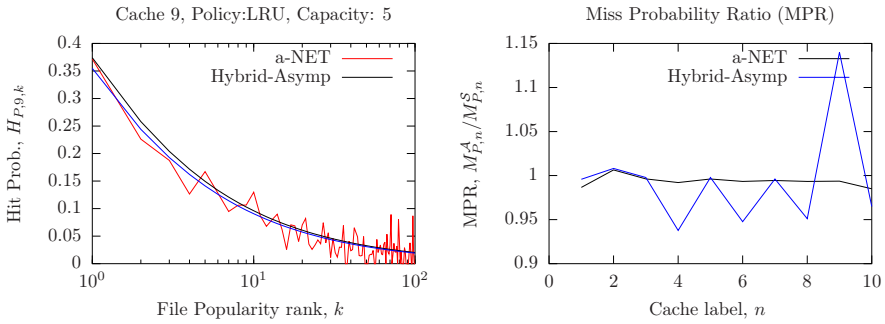


Figure 20: Simulations, a-NET/Poisson, and *Hybrid-Asymp* approximations on the ten caches random network **with exogenous requests at all caches**. $K = 10^2$ and $C = 5$.

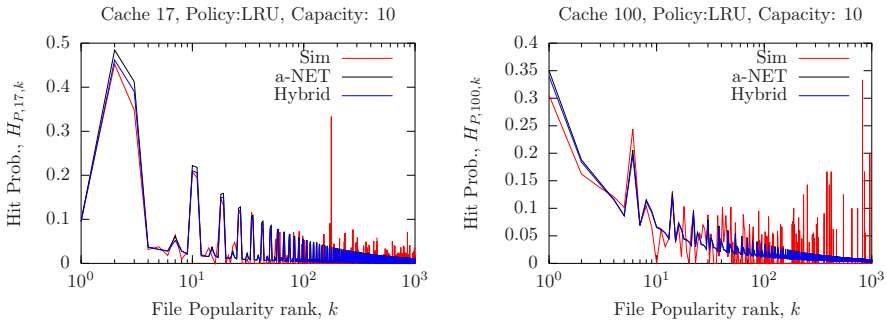


Figure 21: Simulations, Hybrid and a-NET/Poisson approximations on a hundred caches random network **with exogenous requests at all caches**. $K = 10^3$ and $C = 10$.

7 Conclusion

In this paper, we studied performance analysis of general and heterogeneous cache networks where caches are running LRU, FIFO and RND replacement policies under the assumption that request streams are described by renewal processes. Our methodology is built on four independent and extensible building blocks—namely APA, CTA, PMC, and RRS—that provide a step-by-step analysis of cache networks. In fact, these blocks translate the cache network primitives (i.e. caching and routing) i.e the request flow transformations into operations on point processes such as superposition, thinning, and filtering.

We implemented our methodology by first assuming that all request streams are described by Poisson processes. Our simulations show that the Poisson approximation is similar to the existing a-NET model [23] and accurate when the in-degree of a cache is large or in the presence of exogenous Poisson requests on the cache. To handle more networks configurations, we assumed that requests were described by hyper/shifted-exponential renewal processes introduce by Whitt to address the temporal locality in request streams in a better way than Poisson processes. We found that Whitt approximations—namely *Whitt-Interv* and *Whitt-Asymp*—are more accurate for cache networks where requests occurs at edge nodes (e.g. at leaves of a tree). To overtake this limitation and address all network configurations, we proposed the Hybrid approximations that combines Whitt (*Whitt-Interv* or *Whitt-Asymp*) and Poisson approximations. Our experiments show that the *Hybrid-Asymp* approximation (that combines *Whitt-Asymp* and Poisson approximations) provide the best tradeoff accuracy-robustness to predict the file hit probabilities and the per-cache MPRs.

Future work will devoted to an in-depth investigation of the sources of error predictions of the *Hybrid-Asymp* approximation and assess its accuracy with real data traces and network topologies.

References

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

[2] S. Albin. Approximating a point process by a renewal process, II: Superposition arrival processes to queues. *Operations Research*, 32(5), Sept. 1984.

[3] D. S. Berger, P. Gland, S. Singla, and F. Ciucu. Exact analysis of TTL cache networks: The case of caching policies driven by stopping times. *CoRR*, abs/1402.5987, 2014.

[4] H. Che, Y. Tung, and Z. Wang. Hierarchical web caching systems: modeling, design and experimental results. *IEEE JSAC*, 20(7):1305–1314, 2002.

[5] N. Choungmo Fofack. *On models for performance analysis of a core cache network and power save of a wireless access network*. PhD thesis, Feb. 2014.

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

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

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

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

[10] N. Choungmo Fofack, P. Nain, G. Neglia, and D. Towsley. Performance evaluation of hierarchical ttl-based cache networks. *Computer Networks*, (0):–, 2014.

[11] C. Fricker, P. Robert, and J. Roberts. A versatile and accurate approximation for LRU cache performance. *CoRR*, abs/1202.3974, 2012.

[12] M. Gallo, B. Kauffmann, L. Muscariello, A. Simonian, and C. Tanguy. Performance evaluation of the random replacement policy for networks of caches. *Perf. Eval.*,72(0):16-36, 2014.

[13] V. Isham. Dependent thinning of point processes. *J. of Appl. Prob.*, 17(4):987-995, Dec. 1980.

[14] P. R. Jelenkovic and X. Kang. Characterizing the miss sequence of the lru cache. *SIGMETRICS Perform. Eval.*, 36(4):119–121, Aug. 2008.

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

[16] N. Laoutaris. A closed-form method for LRU replacement under generalized power-law demand. *CoRR*, abs/0705.1970, May 2007.

[17] N. Laoutaris, S. Syntila, and I. Stavrakakis. Meta algorithms for hierarchical web caches. In *Proc. 23rd IEEE IPCCC'04*, Phoenix, Arizona, USA, Apr. 2004.

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

[19] V. Martina, M. Garetto, and E. Leonardi. A unified approach to the performance analysis of caching systems. *CoRR*, abs/1307.6702, Sept. 2013.

[20] N. Melazzi, G. Bianchi, A. Caponi, and A. Detti. A general, tractable and accurate model for a cascade of lru caches. *Communications Letters, IEEE*, PP(99):1–4, 2014.

[21] N. Choungmo Fofack and Sara Alouf. Modeling modern DNS caches. In *Proc. ACM ValueTools'13*, Torino, Italy, Dec. 2013.

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

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

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

[25] C. Y. Teresa Lam and J. P. Lehoczky. Superposition of renewal processes. Technical Report TR-89-12, University of Michigan, Ann Arbor, MI 48109-2117, USA, oct 1990.

[26] S. Traverso, M. Ahmed, M. Garetto, P. Giaccone, E. Leonardi, and S. Niccolini. Temporal locality in today's content caching: Why it matters and how to model it. *ACM Computer Communication Review*, 43(5), Oct. 2013.

[27] W. Whitt. Approximating a point process by a renewal process, I: Two basic methods. *Operations Research*, 30(1), 1982.

A Approximation of general point processes

We focus on the two particular renewal processes with an hyper-exponential or a shifted-exponential CDF for inter-arrival times—introduced by Whitt [27] to approximate any general point process, and that we use in this paper to approximate the aggregated request arrival process and the miss process of our caches.

1. If X has an Hyper-exponential CDF with parameters p_1, p_2, θ_1 and θ_2, then
   (a) the mean and the variance of X,
   $$E[X] = p_1\theta_1^{-1} + p_2\theta_2^{-1}, \quad Var[X] = p_1\theta_1^{-2} + p_2\theta_2^{-2} - E[X]^2$$
   (b) CDF of inter-request time X,
   $$F(t) = 1 - (p_1 e^{-\theta_1 t} + p_2 e^{\theta_2 t}),\ p_1 + p_2 = 1,\ t \geq 0$$
   (c) CDF of forward recurrence time,
   $$\hat{F}(t) = 1 - \left(\frac{p_1\theta_1^{-1}}{E[X]} e^{-\theta_1 t} + \frac{p_2\theta_2^{-1}}{E[X]} e^{-\theta_2 t}\right)$$
   (d) Renewal Function associated to $F(t)$,
   $$M(t) = \frac{t}{E[X]} - p_1 p_2 \left(\frac{\theta_1 - \theta_2}{p_1\theta_2 + p_2\theta_1}\right)^2 \left(1 - e^{-(p_1\theta_2 + p_2\theta_1)t}\right)$$
   (e) LST of $F(t)$
   $$F^*(s) = \frac{p_1\theta_1}{\theta_1 + s} + \frac{p_2\theta_2}{\theta_2 + s}$$
2. If X has a Shifted-exponential CDF with parameters τ and θ, then
   (a) the mean and the variance of X,
   $$E[X] = \tau + \theta^{-1}, \qquad Var[X] = \theta^{-2}$$
   (b) CDF of inter-request time X,
   $$F(t) = 1 - e^{-\theta(t-\tau)},\ t \geq \tau$$
   (c) CDF of forward recurrence time,
   $$\hat{F}(t) = \left(1 - \frac{e^{-\theta(t-\tau)}}{1 + \tau\theta}\right)\ \mathbf{1}(t \geq \tau) + \left(\frac{\theta t}{1 + \theta\tau}\right)\ \mathbf{1}(t < \tau)$$
   (d) Renewal Function associated to $F(t)$,
   $$M(t) = \sum_{k=1}^{K} \frac{\gamma\left(k, \theta(t - k\tau)\right)}{\Gamma(k)},\ K = \left\lfloor \frac{t}{\tau} \right\rfloor,$$
   where $\gamma(.)$ is the incomplete Gamma function.
   (e) LST of $F(t)$
   $$F^*(s) = \frac{\theta}{\theta + s} e^{-s\tau}$$

B Fifo

The quantities $L_i^*(0) = L_i(\infty) = M_i(T)$, $L_i^*(s)$ and its derivative $\frac{\mathrm{d}L_i^*(s)}{\mathrm{d}s}\Big|_{s=0} = L_i'^*(0)$ are obtained as follows.

- X_i has an Hyper-exponential CDF,

$$\begin{aligned}
L_i^*(s) &= \frac{1-e^{-sT}}{sE[X_i]} + \frac{p_1p_2(\theta_1-\theta_2)^2}{\delta}\frac{1-e^{-(s+\delta)T}}{s+\delta} \\
L_i^*(0) &= \frac{T}{E[X_i]} + p_1p_2\left(\frac{(\theta_1-\theta_2)}{\delta}\right)^2(1-e^{-\delta T}) \\
L_i'^*(0) &= \frac{-T^2}{2E[X_i]} - \frac{p_1p_2(\theta_1-\theta_2)^2}{\delta}\frac{1-e^{-\delta T}-\delta Te^{-\delta T}}{\delta^2} \\
\delta &= p_1\theta_2 + p_2\theta_1
\end{aligned}$$

- X_i has an Shifted-exponential CDF,

$$\begin{aligned}
L_i^*(s) &= \sum_{k=1}^{K}\left(\frac{\theta e^{-s\tau}}{s+\theta}\right)^k \frac{\gamma\left(k,(s+\theta)(T-k\tau)\right)}{\Gamma(k)} \\
L_i^*(0) &= \sum_{k=1}^{K}\frac{\gamma\left(k,\theta(T-k\tau)\right)}{\Gamma(k)} \\
L_i'^*(0) &= \sum_{k=1}^{K}\frac{(\theta(T-k\tau))^k e^{-\theta(T-k\tau)}}{\theta\Gamma(k)} - \frac{k\gamma\left(k,\theta(T-k\tau)\right)}{\Gamma(k)E[X_i]}, \\
1 \le K &= \left\lfloor\frac{T}{\tau}\right\rfloor
\end{aligned}$$

C Random

Recalling that $\mu = T^{-1}$, $L_i^*(s) = F_i^*(s+\mu)$, $L_i^*(0)$ and $\frac{\mathrm{d}L_i^*(s)}{\mathrm{d}s}\Big|_{s=0} = L_i'^*(0)$ are obtained as follows.

- X_i has an Hyper-exponential CDF,

$$\begin{aligned}
L_i^*(0) &= F_i^*(\mu) = \frac{p_1\theta_1}{\theta_1+\mu} + \frac{p_2\theta_2}{\theta_2+\mu} \\
L_i'^*(0) &= -\frac{p_1\theta_1}{(\theta_1+\mu)^2} - \frac{p_1\theta_1}{(\theta_1+\mu)^2}
\end{aligned}$$

- X_i has an Shifted-exponential CDF,

$$\begin{aligned}
L_i^*(0) &= F_i^*(\mu) = \frac{\theta e^{-\mu\tau}}{\theta+\mu} \\
L_i'^*(0) &= -F_i^*(\mu)(\tau + (\theta+\mu)^{-1})
\end{aligned}$$

D LRU

The quantities $L_i^*(s)$ and its derivative $\frac{\mathrm{d}L_i^*(s)}{\mathrm{d}s}\Big|_{s=0} = L_i^{'*}(0)$ are

- X_i has an Hyper-exponential CDF,

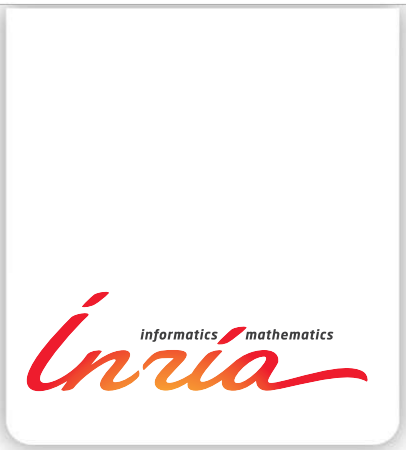
$$\begin{aligned} L_i^*(0) &= F_i(T) = 1 - p_1 e^{-\theta_1 T} - p_2 e^{-\theta_2 T} \\ L_i^{'*}(0) &= -\frac{p_1}{\theta_1}(1 - e^{-\theta_1 T} - \theta_1 T e^{-\theta_1 T}) - \frac{p_2}{\theta_2}(1 - e^{-\theta_2 T} - \theta_2 T e^{-\theta_2 T}) \end{aligned}$$

- X_i has an Shifted-exponential CDF, for $T \geq \tau$

$$\begin{aligned} L_i^*(0) &= F_i(T) = 1 - e^{-\theta(T-\tau)} \\ L_i^{'*}(0) &= -E[X_i](1 - e^{-\theta(T-\tau)}) + (T-\tau)e^{-\theta(T-\tau)} \end{aligned}$$

  with $L^*(0) = 0$ and $L_i^{'*}(0) = 0$ if $T < \tau$.

Contents



RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

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

ISSN 0249-6399