LRU-2 vs 2-LRU: An Analytical Study

Alireza Montazeri
Department of Computer Science
University of Saskatchewan
Saskatoon, Canada
Email: alm164@mail.usask.ca

Nicholas R. Beaton
School of Mathematics and Statistics
University of Melbourne
Melbourne, Australia
nrbeaton@unimelb.edu.au

Dwight Makaroff
Department of Computer Science
University of Saskatchewan
Saskatoon, Canada
Email:makaroff@cs.usask.ca

***Abstract*—Hierarchical caching enables users to obtain content from one of (possibly) many caches between an edge router cache and an origin server, reducing latency/overall network traffic. This can be used in many network architectures, including P2P networks and Content Distribution Networks. Of particular interest are Information Centric Networks (ICNs), which decouple content identifiers from specific network hosts and explicitly consider *universal caching* (collaboration between network routers) as a desirable feature.**

Performance analysis of a large-scale hierarchy of caches requires accurate mathematical models for various cache replacement algorithms. There is no previous study that models LRU-k. This cache replacement algorithm is important since its principle is the basis of recent algorithms such as k-LRU that outperform LRU in many situations. We first model LRU-2 using Che's approximation, as a specific case of LRU-k for $k = 2$. We also extend our model to a hierarchical network of LRU-2 caches. The model is validated analytically and with simulation. The experiments show that the proposed model approximates LRU-2 accurately. LRU-2 and 2-LRU are also compared analytically and with simulations. The comparison between the two algorithms illustrates that 2-LRU outperforms LRU-2.

I. Introduction

One of the key features of P2P Networks and Content Distribution Networks (CDNs) is *universal caching*. The goal of universal caching is moving the most popular data items towards the network edges in order to decrease network traffic, remove the single point-of-failure of servers, thereby improving user-perceived experience. These features are also of interest in emerging Information Centric Networks (ICNs) as well as use in routers within mobile networks [1].

Cache replacement algorithms choose what data items to evict from the cache, when the cache is full, in order to make room for other data items. Algorithm ξ_1 outperforms ξ_2 if the chosen data item for eviction is less likely to be requested in the future, improving the probability of cache hits (i.e. $h_{\xi_1} > h_{\xi_2}$). With a cache capacity of C in an independent Reference Model (IRM) environment (i.e. constant frequency of requests for data items over time), the Least Frequently Used (LFU) algorithm stores the C most popular data items in the cache and provides optimal performance under IRM [2]. LFU evicts the data item with the lowest frequency based on the history recorded. In a non-IRM environment however, Garetto *et al.* show that LFU is not optimal when popularity of data items changes over time [3]. It adapts poorly to temporal locality in practice, caching stale items with past high frequency [4].

Least Recently Used (LRU) is a popular replacement algorithm with low run-time cost. Upon the arrival of data item i, if i is at the lth spot in the LRU-sorted queue, LRU moves i to the head of the queue and other data items, located between the head and $l-1^{th}$ spot, one spot down. If i is not already cached, the data item at the tail of the LRU queue is evicted if required. LRU keeps the most recently accessed data items closer to the head of the queue. LRU however, caches singleton data requests, potentially evicting popular data items that have a temporary lull in requests.

O'Neil *et al.* proposed LRU-k, which modifies LRU to keep track of the last k requests for each data item [5]. They assume $D = \{1, 2, \ldots, i, \ldots, N\}$ as a list of data items, and request arrival for data items specified by a reference string like $r_1, r_2, \ldots, r_t$, where $r_t = i \in D$ means that data item i is requested at time t. Given a reference string $r_1, r_2, \ldots, r_t$, the *backward k-distance* for data item i from time t, shown by $b_t(i, k)$, is the number of references since k^{th} most recent reference for i:

$$b_t(i,k) = \begin{cases} x & \text{if } r_{t-x} = i \text{ , and there are exactly } k \\ & \text{references for } i \text{ in time interval } [t-x, t], \\ \infty & \text{if } i \text{ does not appear } k \text{ times.} \end{cases}$$

LRU-k evicts the data item i with largest $b_t(i, k)$; LRU is used if potential victims with $b_t(i, k) = \infty$ are already cached. LRU-k results in a higher hit ratio compared to LRU, but its implementation complexity is $O(\log C)$ since the meta-data list must be kept sorted on each request arrival [6].

This inspired the design of cache replacement algorithms that produce equivalent hit ratios while eliminating implementation overhead. $O(1)$ algorithms LRU-2Q [6], k-LRU [3] and Adaptive Replacement Cache (ARC) [7] have been proposed, considering both recency and frequency of requests.

Garetto *et al.* proposed k-LRU with identical $k-1$ virtual LRU caches (i.e. storing the references of data items) and one physical LRU cache [3]. Before insertion in the physical cache (indexed k), requests have to go through $k-1$ preceding virtual caches. After receiving a request for a data item, the reference/data item can be stored in cache $l > 1$ only if its reference is already stored in cache $l-1$. The k-LRU algorithm is a generalization of LRU-2Q and ARC. They show that a virtual cache before the LRU cache (2-LRU) provides a huge benefit for small caches in IRM and non-IRM environments.

978-1-5386-4413-3/18/$31.00 ©2018 IEEE

Having an accurate model to calculate the miss rates of replacement algorithms is crucial for performance analysis of large-scale interconnected caches. Previous analysis uses Che's approximation [8], initially proposed for LRU, and subsequently other caching algorithms. Recent theoretical models of different cache replacement algorithms such as FIFO [9], LRU [10], q-LRU [9], LRU-2Q [11] and k-LRU [9], [12], have been developed, but none for LRU-k. We believe this algorithm is important since its main principle is used in the previously mentioned improved replacement policies.

Our main contribution is a mathematical model for LRU-2, leveraging Che's approximation for LRU-2 as a specific case of LRU-k. Values of $k > 2$ are not considered because (1) LRU-3 results in only a slightly higher hit ratio compared to LRU-2 and (2) LRU-3 is less responsive to temporal locality since it needs to observe the reference history of data objects over a longer period of time [5]. As the second contribution, the LRU-2 and 2-LRU algorithms are compared in synthetic and realistic topologies. LRU-2 is compared to 2-LRU since they record the history of references and consider both recency and frequency in their eviction process. Though 2-LRU outperforms LRU-2 in all simulations, the accuracy of our LRU-2 model is superior to the 2-LRU model as Zipf parameter α and cache size increase.

The rest of this paper is organized as follows. We begin with related work in Section II. Section III models LRU-2 for both a single cache and a hierarchy of caches. Our model is validated analytically and with simulation in Section IV. This section also compares LRU-2 and 2-LRU. Finally, Section V concludes the paper and outlines future work.

II. Related Work

Modelling the performance of cache networks is challenging; Dan and Towsley argue that the computational cost to approximate the behaviour of a single Least Recently Used (LRU) cache grows exponentially as a function of cache size and population size [13]. Several models have been proposed that approximate cache performance at an affordable computational cost under IRM. Assuming C and N are the cache and population size of cache respectively, Dan and Towsley proposed an approximate technique with complexity $O(CN)$ for the LRU cache hit probability under IRM [13].

Another approximation for LRU caches under IRM was proposed by Che *et al.* [8]. Che's approximation, recognized to be very accurate [3], [14], has since been used to model other caching algorithms. Garetto *et al.* [3] use Che's technique to model FIFO, q-LRU, Random and k-LRU. They show that q-LRU and k-LRU tend asymptotically to LFU as $q \to 0$ and $k \to \infty$, respectively. An open-form expression to model the hit rate of LRU-2Q was found by Imai [15] and then solved through a recursive algorithm. Boyar's study is the only work that studies LRU-2 and theoretically finds LRU-2's superiority over LRU [16]. They do not provide an analytical model to approximate hit ratios.

In the context of a network of caches, ICN nodes deploy cache replication algorithms because of a limited caching capacity of the set of nodes. A cache replication algorithm determines whether ICN node u should cache a data item upon its arrival at u. ICN node u can make an independent decision on caching the arrived data items. In Leave Copy Everywhere (LCE) replication mechanism for example, ICN node u always stores arrived data items. LCE is easy to implement, but results in lots of redundant copies of a data item in ICN nodes on the delivery path. On the other hand, ICN nodes may collaborate with the other nodes on the delivery path to make replication decision to optimize the placement of a data item on the delivery path. Leave Copy Down (LCD) [17], Move Copy Down (MCD) [18], ProbCache [19] and an age-based cache algorithm proposed by Ming *el al.* [20] are examples of this category.

An analytical investigation of network of caches needs to model the arrival rate of users' requests at intermediate ICN nodes. Psaras *et al.* propose a Markovian approach to approximate the behaviour of a hierarchy of LRU caches under IRM [10]. The Markovian assumptions used in their approach make it difficult to be extended to non-IRM traffic and other policies.

The models proposed by Rosensweig *et al.* [21], Carofiglio *et al.* [22] and Dabirmoghaddam *et al.* [23] rely on cache independence, and that requests at intermediate nodes satisfy the IRM assumptions. This makes the hit ratio calculations easier, but causes prediction error. Rosensweig *et al.* for example, propose a-NET, that approximates the miss rates of data items in a network of LRU caches where LCE is used as the cache replication mechanism [21]. Assuming $\lambda^e_{v,i}$ and $P_{v,i}$ as the exogenous request rate and hit probability respectively for item i at ICN node v, $\lambda'_{v,i}$ as the miss rate for data item i at node v, their a-NET algorithm finds the request rate for item i at ICN node v, $\lambda_{v,i}$, through

$$\lambda_{v,i} = \lambda^e_{v,i} + \sum_{u \in R(v)} \lambda'_{u,i} \quad \text{and,} \tag{1}$$

$$\lambda'_{v,i} = \lambda_{v,i}(1 - P_{v,i}), \tag{2}$$

in which, $R(v)$ is the set of all v's neighbouring ICN nodes from which v may receive a request for i.

During their study, Rosensweig *et al.* identified main potential sources of prediction error that appears between the simulation results and their model: the violation of the IRM (or Poisson) assumption on the miss streams of LRU caches at intermediate ICN nodes [21]. So, (1) will predict an inaccurate approximation of miss rates for i at intermediate ICN node v. Thus, others have used TTL-based models for network of caches to calculate an accurate approximation of caching behaviours [12], [24], [25]. The sophisticated mathematical approach makes these models computationally costly.

We extend Che's LRU approximation for LRU-2. For the sake of simplicity, Rosensweig's mechanism is used to calculate the hit ratio in a hierarchy of caches in which 2-LRU/LRU-

2 and LCE are used as replacement and replication algorithms, respectively.

III. Modelling LRU-2

A. Che's LRU Approximation

Requests for item i arrive as a Poisson process with rate λ_i. If new item i is requested, the eviction policy inserts i at the expense of the last item. Che *et al.* devise a mathematically simpler model to approximate LRU. Instead of a fixed size cache, in their model a fixed time τ_i is associated with item i, and i is kept in the cache if time less than τ_i has elapsed since its most recent request. They assume that τ_i can be taken to be deterministic and independent of i; this assumption has been shown to be valid by Fricker *et al.* [14] with a Zipf popularity distribution. Fricker *et al.* show that (1) the coefficient of variation of τ_i vanishes as the cache size grows, and (2) $\tau_i \approx \tau_j$ $(i \neq j)$ when the catalogue is sufficiently large.

We thus set all τ_i equal to τ. The time-average probability P_i that data item i is in the cache is then

$$P_i(\tau) = 1 - e^{-\lambda_i \tau}. \tag{3}$$

To get the best approximation to regular LRU with cache size C, we want the expected number of items in the cache to be C. The value of τ should thus satisfy

$$\sum_{j=1}^{N} P_j(\tau) = \sum_{j=1}^{N} \left(1 - e^{-\lambda_j \tau}\right) = C. \tag{4}$$

Now consider item i, arriving as a Poisson process with rate λ_i, at times $t_1, t_2, t_3 \ldots$; let $u_l = t_l - t_{l-1}$ be the time period between l^{th} and $(l-1)^{th}$ requests. Assume item i was requested at time $t = 0$, and at that point it entered (or re-entered) the cache. Furthermore, say $n \geq 1$ is the smallest value such that $u_n > \tau$. We wish to know the expectation of the sum $S = u_1 + u_2 + \cdots + u_n$, interpreted as the expected time between two consecutive cache misses for item i. This can be computed as follows: we must sample u_1, with expected value $\frac{1}{\lambda_i}$. With probability $e^{-\lambda_i \tau}$ we terminate there, otherwise we sample again. What happens after that is independent of u_1, and so has expected sum $\mathbb{E}[S]$. Hence

$$\mathbb{E}[S] = \frac{1}{\lambda_i} + (1 - e^{-\lambda_i \tau})\mathbb{E}[S]. \tag{5}$$

The miss rate expectation λ_i' for item i is then defined to be

$$\lambda_i' = \frac{1}{\mathbb{E}[S]} = \lambda_i e^{-\lambda_i \tau}. \tag{6}$$

B. Garetto's and Gast's Models for 2-LRU

Assuming τ_v/τ_p for the virtual/physical caches and requests for data item i at both the virtual and physical caches arrive according to a Poisson process of rate λ_i, Garetto *et al.* [3] find the approximate value of item i's hit probability to satisfy

$$[P_i(\tau_v, \tau_p) = (1 - e^{-\lambda_i \tau_p})[P_i + (1 - e^{-\lambda_i \tau_v})(1 - P_i)]. \tag{7}$$

However, the arrival process of requests for data item i at the physical cache is an ON-OFF modulated Poisson process. In the OFF phase, no request for item i is forwarded to the physical cache since item i's pointer is not stored in the virtual cache. In the ON phase however, a request for item i is forwarded to the physical cache. Gast's model for 2-LRU (Eq. 11) [12] considers this ON-OFF process and calculates the hit probability as follows:

$$P_i(\tau_v, \tau_p) = \frac{(1 - e^{-\lambda_i \tau_v})(1 - e^{-\lambda_i \tau_p})}{1 - e^{-\lambda_i \tau_v} + e^{-\lambda_i \tau_p}}. \tag{8}$$

C. The LRU-2 Model

To adapt Che's approximation for LRU-2, we redefine the cache to store item i if less than τ_i has elapsed since the *second-most recent* request. The arrival process of item i is presented in Figure 1. We set t_0 to be a time at which a cache miss occurs and causes the system to put item i into the cache. Cache hits continue to occur at $t_1, t_2, \ldots$, and in general a request for i at time t_l is a cache hit *iff* $t_l - t_{l-2} < \tau_i$.

We then set n such that t_n is the first time after t_0 at which a cache miss occurs (because $t_n - t_{n-2} > \tau_i$). Then set $t_i' = t_{n+i}$. Cache misses continue to occur at $t_1', t_2', \ldots$. Finally, m is defined such that t_m' is the first time after t_0' at which a request results in item i being placed back into the cache (because $t_m' - t_{m-2}' < \tau_i$).

There is one cache miss in interval $(t_0, t_n]$, while all requests for item i end in a cache miss in the interval $(t_0', t_m']$. The expected number of misses in the interval $[t_0, t_m']$ is thus $\lambda_i(t_m' - t_0')$, and we define the expected miss rate λ_i' to be

$$\lambda_i' = \frac{\lambda_i \mathbb{E}[t_m' - t_0']}{\mathbb{E}[t_m' - t_0]} = \frac{\lambda_i \mathbb{E}[t_m' - t_0']}{\mathbb{E}[t_m' - t_0'] + \mathbb{E}[t_n - t_0]}. \tag{9}$$

1) Calculating τ_i: As with LRU, we let $u_l = t_l - t_{l-1}$. For item i, $\{u_1 + u_2, u_2 + u_3 \ldots\}$ are identically distributed Erlang random variables with pdf of $f(t, \lambda_i) = \lambda_i^2 t e^{-\lambda_i t}$. The time-average probability P_i that item i is in the cache is then given by

$$P_i(\tau_i) = 1 - e^{-\lambda_i \tau_i}(1 + \lambda_i \tau_i). \tag{10}$$

We again assume that $\tau_i = \tau$ is independent of i [8]. To model LRU-2 with a cache size C, we want τ to satisfy

$$\sum_{j=1}^{N} P_j(\tau) = \sum_{j=1}^{N} \left(1 - e^{-\lambda_j \tau}(1 + \lambda_j \tau)\right) = C. \tag{11}$$

2) Calculating t_n: For item i, we now have a sequence of exponential random variables $u_1, u_2, u_3 \ldots$ and we terminate when two consecutive values add to more than τ. If $u_{n-1} + u_n$ is the first such pair, define $S_1 = u_0 + \cdots + u_n$. However, with the setup described above, it is not enough to know that item i enters the cache at time t_0: what happens after this *also depends on* $u_0 = t_0 - t_{-1}$. We must therefore take the distribution of u_0 into account.

We first determine the distribution of u_{-1}, initially conditioning only on $u_{-1} + u_{-2} > \tau$ (because the request at t_{-1} was

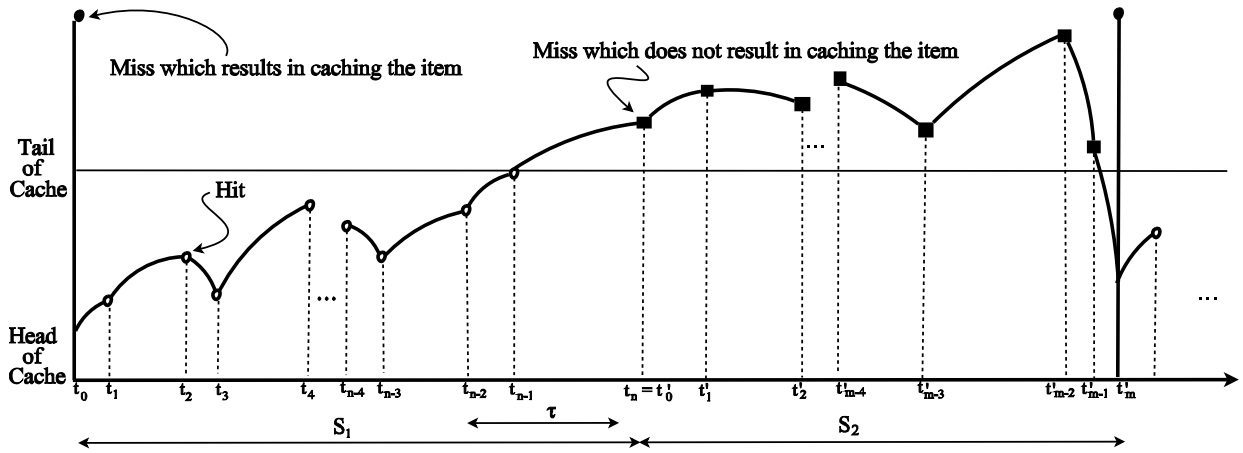


Fig. 1: Arrival processes of data item i at an LRU-2 cache.

a cache miss). Write $f^*_{u_{-1}}(u)$ for this conditional distribution. By Bayes' theorem, it satisfies

$$f^*_{u_{-1}}(u) = \frac{\lambda_i e^{-\lambda_i u}\mathbb{P}[u_{-1}+u_{-2}>\tau \,|\, u_{-1}=u]}{\mathbb{P}[u_{-1}+u_{-2}>\tau]} = \begin{cases} \dfrac{\lambda_i}{1+\tau\lambda_i} & u<\tau \\ \dfrac{\lambda_i e^{\lambda_i(\tau-u)}}{1+\tau\lambda_i} & u>\tau, \end{cases} \tag{12}$$

where the probabilities in the numerator and denominator come from exponential and Erlang distributions respectively, and have thus been computed exactly.

Given the distribution of u_{-1}, we can get the distribution $f^\dagger_{u_0}(u)$ of u_0 in the same way, now conditioning on $u_0+u_{-1}<\tau$ (because the request at t_0 results in i entering the cache):

$$f^\dagger_{u_0}(u) = \frac{\lambda_i e^{-\lambda_i u}\mathbb{P}[u_0+u_{-1}<\tau \,|\, u_0=u]}{\mathbb{P}[u_0+u_{-1}<\tau]} = \frac{\lambda_i^2 e^{-\lambda_i u}(\tau-u)}{\lambda_i\tau+e^{-\lambda_i\tau}-1} \qquad \text{for } 0<u<\tau. \tag{13}$$

This time the numerator (denominator) uses the distribution of (12) (convolution of (12) with an exponential distribution).

We now condition on the value of u_0. Given $u_0=x$, let $S_1(x)$ be the time until the next cache miss. We first sample u_1 with expected value $\frac{1}{\lambda_i}$. If $x>\tau$, we terminate there, regardless of the value of u_1. If $x<\tau$, then with probability $e^{-\lambda_i(\tau-x)}$ we also terminate at u_1. Otherwise, we must keep going, but now what happens afterwards depends on u_1. We must integrate over the possible values of u_1, to obtain

$$\mathbb{E}[S_1(x)] = \frac{1}{\lambda_i} + \int_0^{\tau-x} \lambda_i e^{-\lambda_i u}\mathbb{E}[S_1(u)]du. \tag{14}$$

Let $F(x)=\mathbb{E}[S_1(x)]$, and differentiate w.r.t. x to obtain

$$F'(x) = -\lambda_i e^{-\lambda_i(\tau-x)}F(\tau-x). \tag{15}$$

Solving (15), as shown in Appendix A, results in

$$\mathbb{E}[S_1(x)] = \frac{\frac{1}{\sqrt{\mu_1}}e^{\mu_1(x-\frac{\tau}{2})} - \frac{1}{\sqrt{\mu_2}}e^{\mu_2(x-\frac{\tau}{2})}}{\lambda_i\left(\frac{1}{\sqrt{\mu_1}}e^{\mu_1\frac{\tau}{2}} - \frac{1}{\sqrt{\mu_2}}e^{\mu_2\frac{\tau}{2}}\right)} \tag{16}$$

in which $\mu_{1,2}$ depend on λ_i and τ and are defined in Appendix A (specifically (31)). Finally, to compute $\mathbb{E}[t_n-t_0]$, we combine (16) (conditioned on $u_0=x$) with (13) (the distribution of u_0), shown in (17).

3) Calculating t'_m: Now say item i was requested at time $t=t'_0$, and at that point it was not in the cache. It then has requests at times $t'_1, t'_2, t'_3, \ldots, t'_m$ with m the smallest value such that $t'_m - t'_{m-2} = u'_m + u'_{m-1} < \tau$. We again have a sequence of exponential random variables $u'_1, u'_2, u'_3, \ldots$, but this time we terminate when two consecutive values add to *less than* τ. As before, what happens after t'_0 also depends on u'_0, so we must calculate its distribution.

We first obtain the distribution of u'_{-1}, conditioned on $u'_{-1}+u'_{-2}<\tau$. By Bayes' theorem, this conditional distribution is

$$f^*_{u'_{-1}}(u) = \frac{\lambda_i e^{-\lambda_i u}\mathbb{P}[u_{-1}+u_{-2}<\tau \,|\, u_{-1}=u]}{\mathbb{P}[u_{-1}+u_{-2}<\tau]} = \frac{\lambda_i(e^{-\lambda_i u}-e^{-\lambda_i\tau})}{1-(1+\lambda_i\tau)e^{-\lambda_i\tau}} \qquad \text{for } 0<u<\tau. \tag{18}$$

Then, taking this distribution into account, we compute the distribution of u'_0 conditioned on $u'_{-1}+u_0>\tau$:

$$f^\dagger_{u'_0}(u) = \frac{\lambda_i e^{-\lambda_i u}\mathbb{P}[u_0+u_{-1}>\tau \,|\, u_0=u]}{\mathbb{P}[u_0+u_{-1}>\tau]} = \begin{cases} \dfrac{\lambda_i e^{-\lambda_i u}(-\lambda_i u+e^{\lambda_i u}-1)}{\lambda_i\tau+e^{-\lambda_i\tau}-1} & u<\tau \\ \dfrac{\lambda_i e^{-\lambda_i u}(-\lambda_i\tau+e^{\lambda_i\tau}-1)}{\lambda_i\tau+e^{-\lambda_i\tau}-1} & u>\tau. \end{cases} \tag{19}$$

We now condition on u'_0. Given $u'_0=y$, let $S_2(y)$ be the time that elapses after t'_0 until item i re-enters the cache. We sample u'_1 with expected value $\frac{1}{\lambda_i}$. If $y>\tau$, we continue and sample u'_2, regardless of u'_1's value (see Appendix B). If $y<\tau$, with probability $1-e^{-\lambda_i(\tau-y)}$ the process terminates after u'_1. Otherwise we keep going, but now what happens after depends on the value of u'_1, so we integrate over the possible values, to get

$$\mathbb{E}[S_2(y)] = \frac{1}{\lambda_i} + \int_{\tau-y}^{\infty} \lambda_i e^{-\lambda_i u}\mathbb{E}[S_2(u)]du. \tag{20}$$

$$\mathbb{E}[t_n - t_0] = \int_0^\tau f^\dagger_{u_0}(u)\mathbb{E}[S_1(u)]du = \frac{e^{\frac{3}{2}\lambda_i\tau}\Big(e^{-\lambda_i\tau}(\mu_1^{\frac{3}{2}}e^{\mu_1\frac{\tau}{2}} - \mu_2^{\frac{3}{2}}e^{\mu_2\frac{\tau}{2}}) + \mu_2^{\frac{3}{2}}e^{-\mu_2\frac{\tau}{2}}(1-\mu_1\tau) - \mu_1^{\frac{3}{2}}e^{-\mu_1\frac{\tau}{2}}(1-\mu_2\tau)\Big)}{\lambda_i^2(\lambda_i\tau + e^{-\lambda_i\tau} - 1)(\sqrt{\mu_2}e^{\mu_1\frac{\tau}{2}} - \sqrt{\mu_1}e^{\mu_2\frac{\tau}{2}})}. \quad (17)$$

Let $H(y) = \mathbb{E}[S_2(y)]$, and differentiate w.r.t. y to get

$$H'(y) = \lambda_i e^{-\lambda_i(\tau - y)}H(\tau - y). \quad (21)$$

Solving (21) (Appendix B), results in

$$\mathbb{E}[S_2(y)] = \begin{cases} \dfrac{H(y)}{\lambda_i(1 - e^{-\tau\lambda_i} - Q)H(\tau)} & y < \tau \\ \dfrac{1}{\lambda_i(1 - e^{-\tau\lambda_i} - Q)} & y \geq \tau, \end{cases} \quad (22)$$

where $H(y)$ and Q are defined in (34) and (37) respectively.

Finally, to compute $\mathbb{E}[t'_m - t'_0]$, we combine (22) (which is conditioned on $u'_0 = y$) with (19) (the distribution of u'_0), shown in (23).

D. Hierarchy of Caches

Suppose cache c has a set of child caches denoted by L. Then let $\lambda_{i,e}$ be the exogenous requests coming from users directly connected to cache c, and $\lambda'_{i,l}$ be item i's miss rate coming from child cache $l \in L$. We then use (14), (17) and (23) to obtain τ_c, $\mathbb{E}[t_{n,c} - t_{0,c}]$ and $\mathbb{E}[t'_{m,c} - t'_{0,c}]$ respectively, where Rosensweig *et al.* define $\lambda_{i,c} = \lambda_{i,e} + \sum_{l \in L}\lambda'_{i,l}$ as the overall request rate for item i at cache c [21].

We do not model a DAG for the cache organization in which an edge or intermediate node could have more than one parent cache. This is done because the intent of universal caching is to provide replication and reduction of latency between the origin server and the requesting user. Direct connections between siblings would lead to cycles in the cache network, complicating the model and is beyond the scope of the current work.

IV. Model Validation/Insights

In this section, we first validate previously derived analytical expressions against simulations using ccnSim [26], showing the accuracy of our models based on different system/traffic parameters. We study LRU-2 and 2-LRU in a benchmark topology and further in four realistic topologies.

All caches have the same size, and edge caches have identical exogenous request patterns, modelled as a Poisson arrival distribution with $\lambda_e = \sum_{\forall i}\lambda_{i,e} = 2$. Item i's request probability (p_i), follows a Zipf distribution with α ($p_i \propto \frac{1}{i^\alpha}$) and $\lambda_{i,e} = \lambda_e p_i$. We use (9) to evaluate our model. The simulations run for 9×10^6 seconds (104 days) of simulation time and $N = 20000$ so the unpopular data items have a non-negligible request frequency.

We consider two scenarios:

- Scenario 1: $C = 200$, $\alpha \in \{0.8, 1.0, 1.2, 1.4\}$, since there is no consensus in the research community on the value of α. Researchers found values of α between 0.6 [27] and 2 [22] for Zipf distributions for different types of data items.
- Scenario 2: $\alpha = 1.0$, $C \in \{200, 500, 1000, 2000\}$.

A. Benchmark Evaluation of the LRU-2 Model

We consider a simple topology of a binary tree with three caches. Only the two edge caches $\{c_1, c_2\}$ have user requests. We compare the miss rate per data item measured by simulations and estimated by the proposed model.

In Scenario 1 (Figure 2a), we calculate τ_{c_1} and τ_{c_2} using (11) to get the miss rate for each item. For edge caches, $\lambda_{i,c_1} = \lambda_{i,c_2} = \lambda_{i,e}$, and $\tau_{c_1} = \tau_{c_2} = 969, 1056, 1554, 2963$ for the different values of α, respectively. Then, we calculate the expectations of t_{n,c_1} and t_{n,c_2} using (17) and t'_{m,c_1} and t'_{m,c_2} using (23), to find λ'_{i,c_1} and λ'_{i,c_2} through (9). The request rate for item i at the root cache r is calculated as in Section III-D which results in $\lambda_{i,r} = \lambda'_{i,c_1} + \lambda'_{i,c_2}$ with no exogenous requests at the root. Having τ_r calculated from $\lambda_{i,r}$, we now calculate item i's miss rates at the root (Figure 2b). The proposed model for LRU-2 provides an excellent approximation of the cache miss rates. Scenario 2 is evaluated in Figure 3 with similar results.

B. LRU-2 vs 2-LRU Evaluation

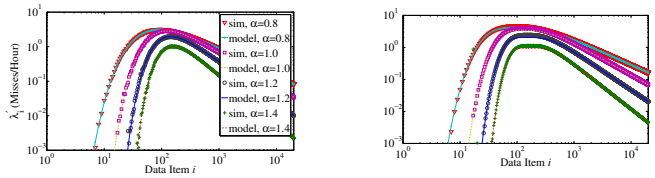
We next compare LRU-2 and 2-LRU with respect to miss rate performance at each level of the cache hierarchy. LRU-2 simulation results are removed since they match the LRU-2 model precisely. In addition, Gast's model (8) [12] for 2-LRU is used since it calculates a more accurate approximation for 2-LRU compared to Garetto's model (7) [3].

The comparison between 2-LRU and LRU-2 for edge nodes in Scenario 1 is demonstrated in Figure 4. The x-axis is cut off at $2C$, since all the remaining data items have 100% miss ratios for both models. The miss rates for popular items is very low as they are almost always in the cache.

This figure shows that the 2-LRU model has a lower miss rate prediction than LRU-2 in all cases, although 2-LRU model does not always match the simulations. The LRU-2 and 2-LRU predictions converge as α increases. In addition, the 2-LRU model underestimates the miss rate as α increases. The same general behaviour at the root cache is observed as shown in Figure 5. The comparison for Scenario 2 is depicted in Figures 6 and 7. There is a close match between the 2-LRU simulation and the LRU-2 model at larger cache sizes; the 2-LRU model slightly underestimates the miss rates.

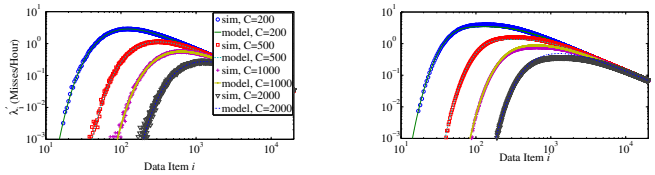
Thus, we can conclude that 2-LRU outperforms LRU-2 for single caches. Moreover, Gast's LRU-2 approximation (8) underestimates 2-LRU miss rates for larger α and cache size, while the proposed model for LRU-2 calculates a better approximation of 2-LRU behaviour in those situations.

$$\mathbb{E}[t'_m - t'_0] = \int_0^{\tau} f^{\dagger}_{u'_0}(u)\mathbb{E}[S_2(u)]du$$
$$= \frac{e^{\lambda_i \frac{\tau}{2}}\left(\sqrt{u_2}(\mu_1\lambda_i\tau - \mu_2)e^{\mu_1\frac{\tau}{2}} + \sqrt{\mu_1}(\mu_2\lambda_i\tau - \mu_1)e^{\mu_2\frac{\tau}{2}}\right) + \lambda_i^2 e^{-\lambda_i\tau}H(\tau)(-\lambda_i\tau + e^{\lambda_i\tau} - 1)}{\lambda_i^3 H(\tau)(\lambda_i\tau + e^{-\lambda_i\tau} - 1)(1 - e^{-\lambda_i\tau} - Q)}. \quad (23)$$



(a) Edge caches (b) Root cache

Fig. 2: LRU-2, Scenario 1 (log-log scale).



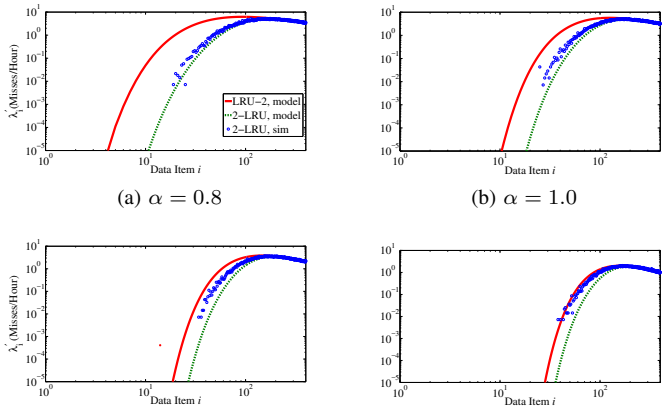
(a) Edge caches (b) Root cache

Fig. 3: LRU-2, Scenario 2 (log-log scale).

C. Realistic Topologies

Apart from k-array tree topologies [10], [19], [23], ISP topologies are used to evaluate the performance of caching in ICNs to have a more realistic study of the behaviour of in-network caching in ICNs. Four publicly available ISP topologies, shown in Table I are used in this paper, as they are also used in other studies [28], [29].

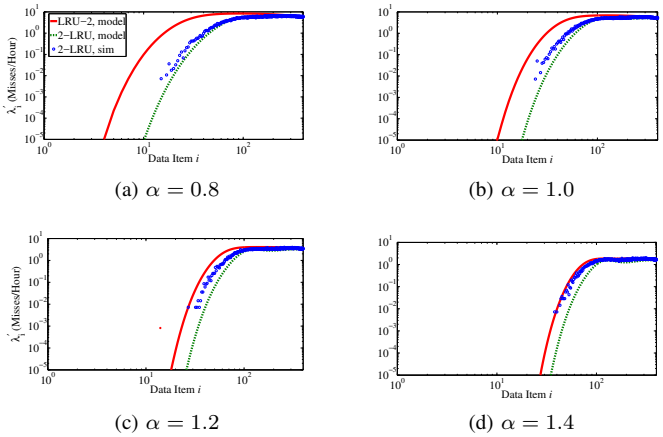
In these topologies, only edge caches are directly connected to users. The intermediate caches receive endogenous requests



(a) $\alpha = 0.8$ (b) $\alpha = 1.0$

(c) $\alpha = 1.2$ (d) $\alpha = 1.4$

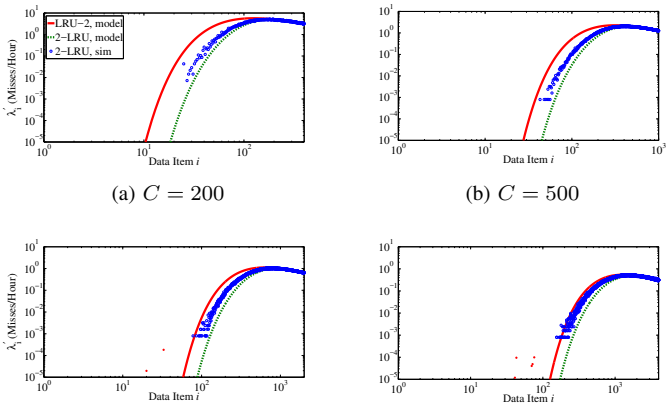
Fig. 4: LRU-2 *vs* 2-LRU, scenario 1, edge



(a) $\alpha = 0.8$ (b) $\alpha = 1.0$

(c) $\alpha = 1.2$ (d) $\alpha = 1.4$

Fig. 5: LRU-2 *vs* 2-LRU, scenario 1, root



(a) $C = 200$ (b) $C = 500$

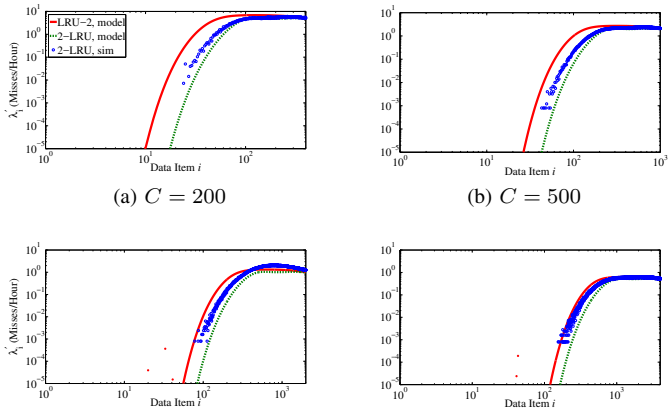
(c) $C = 1000$ (d) $C = 2000$

Fig. 6: LRU-2 *vs* 2-LRU, scenario 2, edge

from their children caches, with a single source node as the gateway. We create a tree rooted at the source node and compute the overall hit probability of the network with λ_e, and α as before and various cache sizes. The overall hit probability of the network is calculated as

$$\Big(\sum_{\forall c}\sum_{\forall i}(\lambda_{c,i} - \lambda'_{c,i})\Big)/\Big(\sum_{\forall c}\sum_{\forall i}\lambda_{c,i}\Big).$$

Table II gives hit probability predictions for 2-LRU and LRU-2 in Geant topology for Gast's 2-LRU model and our LRU-2 model respectively, and the corresponding simulations. 2-LRU provides a slightly higher overall hit ratio in both simulation and modelling. Gast's 2-LRU model has an er-



(a) $C = 200$ (b) $C = 500$

(c) $C = 1000$ (d) $C = 2000$

Fig. 7: LRU-2 *vs* 2-LRU, scenario 2, root

TABLE I: Specification of topologies.

name	*inter-nodes*	*edge-nodes*	*depth*	*max degree*	*average degree*
Level3	5	41	5	29	9.00
Dtelecom	7	61	4	52	9.57
Tiger	12	10	5	4	1.75
Geant	12	10	6	4	1.75

ror between $(+0.4\%, +5.2\%)$ for the overall hit probability, while the proposed LRU-2 model has an error between $(-0.5\%, 6.0\%)$. The LRU-2 model is worse for small α and small caches. The results for Tiger (Table III), DTelecom (Table IV) and Level3 (Table V) confirm this difference.

TABLE II: LRU-2 vs 2-LRU, Geant Topology.

C	2-LRU		LRU-2		
	Sim	Model (err)	Sim	Model (err)	vs. 2-LRU Sim
			$\alpha = 0.8$		
100	0.095	0.096 (+1.1%)	0.084	0.089 (+6.0%)	-6.3%
500	0.169	0.173 (+2.4%)	0.156	0.165 (+5.8%)	-2.4%
1000	0.220	0.227 (+3.2%)	0.206	0.218 (+5.8%)	-0.9%
2000	0.293	0.305 (+4.1%)	0.281	0.297 (+5.7%)	+1.4%
4000	0.407	0.428 (+5.2%)	0.398	0.422 (+6.0%)	+3.7%
			$\alpha = 1.0$		
100	0.223	0.226 (+1.4%)	0.210	0.215 (+2.4%)	-3.6%
500	0.346	0.352 (+1.7%)	0.331	0.341 (+3.0%)	-1.5%
1000	0.416	0.424 (+1.9%)	0.402	0.412 (+2.5%)	-1.0%
2000	0.503	0.515 (+2.4%)	0.491	0.505 (+2.9%)	+0.4%
4000	0.616	0.632 (+2.6%)	0.608	0.622 (+2.3%)	+1.0%
			$\alpha = 1.2$		
100	0.442	0.447 (+1.1%)	0.429	0.437 (+1.9%)	-1.1%
500	0.608	0.614 (+1.0%)	0.596	0.605 (+1.5%)	-0.5%
1000	0.682	0.689 (+1.0%)	0.672	0.673 (+0.2%)	-1.3%
2000	0.756	0.764 (+1.1%)	0.749	0.753 (+0.5%)	-0.4%
4000	0.832	0.841 (+1.1%)	0.828	0.831 (+0.4%)	-0.1%
			$\alpha = 1.4$		
100	0.687	0.692 (+0.7%)	0.678	0.684 (+0.9%)	-0.4%
500	0.834	0.838 (+0.5%)	0.829	0.825 (-0.5%)	-1.1%
1000	0.881	0.885 (+0.5%)	0.877	0.877 (+0.0%)	-0.5%
2000	0.919	0.923 (+0.4%)	0.916	0.918 (+0.2%)	-0.1%
4000	0.949	0.954 (+0.5%)	0.948	0.950 (+0.2%)	+0.1%

For larger cache sizes, the 2-LRU model continues to overestimate and the LRU-2 model is stable in nearly all cases. For smaller values of α, both algorithms are relatively less accurate. The last column in Tables II to V shows the fit between the proposed model for LRU-2 and the simulation

TABLE III: LRU-2 vs 2-LRU, Tiger Topology.

C	2-LRU		LRU-2		
	Sim	Model (err)	Sim	Model (err)	vs. 2-LRU Sim
			$\alpha = 0.8$		
100	0.102	0.103 (+1.0%)	0.091	0.096 (+5.5%)	-5.9%
500	0.181	0.184 (+1.7%)	0.167	0.175 (+4.8%)	-3.3%
1000	0.234	0.239 (+2.1%)	0.220	0.230 (+4.6%)	-1.7%
2000	0.308	0.319 (+3.6%)	0.296	0.309 (+4.4%)	+0.3%
4000	0.423	0.440 (+4.0%)	0.414	0.430 (+3.9%)	+1.7%
			$\alpha = 1.0$		
100	0.239	0.241 (+0.9%)	0.225	0.230 (+2.2%)	-3.8%
500	0.366	0.371 (+1.4%)	0.351	0.359 (+2.3%)	-1.9%
1000	0.437	0.444 (+1.6%)	0.423	0.432 (+2.1%)	-1.1%
2000	0.523	0.533 (+1.9%)	0.512	0.523 (+2.2%)	+0.0%
4000	0.633	0.647 (+2.2%)	0.626	0.636 (+1.6%)	+0.5%
			$\alpha = 1.2$		
100	0.464	0.468 (+0.9%)	0.451	0.458 (+1.6%)	-1.3%
500	0.629	0.634 (+0.8%)	0.618	0.624 (+1.0%)	-0.8%
1000	0.700	0.706 (+0.9%)	0.691	0.690 (-0.1%)	-1.4%
2000	0.771	0.778 (+0.9%)	0.764	0.766 (+0.3%)	-0.7%
4000	0.842	0.874 (+3.8%)	0.839	0.842 (+0.4%)	+0.0%
			$\alpha = 1.4$		
100	0.706	0.711 (+0.7%)	0.697	0.699 (+0.3%)	-1.0%
500	0.846	0.850 (+0.5%)	0.841	0.836 (-0.6%)	-1.2%
1000	0.890	0.893 (+0.3%)	0.886	0.886 (+0.0%)	-0.5%
2000	0.925	0.929 (+0.4%)	0.923	0.924 (+0.1%)	-0.1%
4000	0.953	0.957 (+0.4%)	0.952	0.954 (+0.2%)	+0.1%

results of 2-LRU. As in the synthetic cases, the LRU-2 model with large cache and large α matches the 2-LRU simulations, accurate to 2 (sometimes 3) decimal places.

For the Dtelecom topology, for example (Table IV), the proposed model for LRU-2 approximates the hit ratio of the 2-LRU algorithm with 7.0% error (absolute value of error) for $\alpha = 0.8$ and $C = 100$ - larger than the 2.6% error of Gast's model. The error of the LRU-2 model's prediction for 2-LRU however drops to 1.8% for $C = 4000$ - smaller than the 4.9% error for Gast's model. One can also say that the hit ratio for 2-LRU estimated by LRU-2 gets more accurate as α gets larger; i.e. for $C = 100$ and Level3 topology for example, the error (absolute value of error) of the LRU-2 prediction for 2-LRU decreases from 4.1% to 0.3% as α moves from 0.8 to 1.4. The inaccuracy at smaller α values is due to intermediate node request patterns violating the IRM assumption to a greater degree; this evaluation is part of future work.

V. Conclusion/Future Work

In this paper, we proposed a mathematical model for LRU-2, extending Che's approximation, as a specific case of LRU-k for $k = 2$. The experiments validated the LRU-2 model with respect to the miss rate of data items for LRU-2 caching algorithms. Simulation results show that although 2-LRU outperforms LRU-2 (both in single and hierarchical caches), Gast's 2-LRU model underestimates the miss rate as either Zipf parameter (α) or cache size increases. Additionally, the proposed model for LRU-2 calculates a better approximation of 2-LRU behaviour as either α or cache size increases.

We focused on finding an analytical model for LRU-2 under IRM. Studying our model for non-stationary requests is part of our future work in addition to ICNs with off-path caching. Off-path caching is a form of peer-based cache connections,

TABLE IV: LRU-2 vs 2-LRU, Dtelecom Topology.

C	2-LRU		LRU-2		
	Sim	Model (err)	Sim	Model (err)	vs. 2-LRU Sim
$\alpha = 0.8$					
100	0.115	0.118 (+2.6%)	0.101	0.107 (+5.9%)	-7.0%
500	0.200	0.207 (+3.5%)	0.184	0.193 (+4.9%)	-3.5%
1000	0.256	0.266 (+3.9%)	0.240	0.252 (+5.0%)	-1.6%
2000	0.335	0.350 (+4.5%)	0.321	0.335 (+4.4%)	+0.0%
4000	0.453	0.475 (+4.9%)	0.443	0.461 (+4.1%)	+1.8%
$\alpha = 1.0$					
100	0.264	0.269 (+1.9%)	0.248	0.255 (+2.8%)	-3.4%
500	0.398	0.405 (+1.8%)	0.381	0.390 (+2.4%)	-2.0%
1000	0.470	0.480 (+2.1%)	0.455	0.465 (+2.2%)	-1.1%
2000	0.557	0.570 (+2.3%)	0.545	0.556 (+2.0%)	-0.2%
4000	0.664	0.681 (+2.6%)	0.657	0.668 (+1.7%)	+0.6%
$\alpha = 1.2$					
100	0.499	0.506 (+1.4%)	0.484	0.491 (+1.5%)	-1.6%
500	0.661	0.668 (+1.1%)	0.649	0.656 (+1.1%)	-0.8%
1000	0.729	0.736 (+1.0%)	0.720	0.722 (+0.3%)	-1.0%
2000	0.795	0.803 (+1.0%)	0.789	0.787 (-0.3%)	-1.0%
4000	0.860	0.870 (+1.2%)	0.857	0.859 (+0.2%)	-0.1%
$\alpha = 1.4$					
100	0.735	0.741 (+0.8%)	0.726	0.718 (-1.1%)	-2.3%
500	0.864	0.868 (+0.5%)	0.859	0.850 (-1.1%)	-1.6%
1000	0.903	0.907 (+0.4%)	0.900	0.896 (-0.4%)	-0.8%
2000	0.934	0.938 (+0.4%)	0.932	0.934 (+0.2%)	+0.0%
4000	0.958	0.963 (+0.5%)	0.958	0.958 (+0.0%)	+0.0%

TABLE V: LRU-2 vs 2-LRU, Level3 Topology.

C	2-LRU		LRU-2		
	Sim	Model (err)	Sim	Model (err)	vs. 2-LRU Sim
$\alpha = 0.8$					
100	0.098	0.102 (+4.08%)	0.085	0.094 (+10.6%)	-4.1%
500	0.173	0.183 (+5.8%)	0.158	0.172 (+8.9%)	-0.6%
1000	0.225	0.239 (+6.2%)	0.209	0.227 (+8.6%)	+0.9%
2000	0.299	0.321 (+7.4%)	0.284	0.308 (+8.5%)	+3.0%
4000	0.415	0.448 (+8.0%)	0.403	0.430 (+6.7%)	+3.6%
$\alpha = 1.0$					
100	0.231	0.237 (+2.6%)	0.214	0.226 (+5.6%)	-2.2%
500	0.355	0.366 (+3.1%)	0.338	0.353 (+4.4%)	-0.6%
1000	0.426	0.440 (+3.3%)	0.410	0.427 (+4.2%)	+0.2%
2000	0.514	0.533 (+3.7%)	0.500	0.520 (+4.0%)	+1.2%
4000	0.627	0.652 (+4.0%)	0.618	0.640 (+3.6%)	+2.1%
$\alpha = 1.2$					
100	0.454	0.462 (+1.8%)	0.438	0.449 (+2.5%)	-1.1%
500	0.619	0.629 (+1.6%)	0.606	0.617 (+1.8%)	-0.3%
1000	0.692	0.703 (+1.6%)	0.681	0.690 (+1.3%)	-0.3%
2000	0.765	0.777 (+1.6%)	0.757	0.763 (+0.8%)	-0.3%
4000	0.839	0.852 (+1.6%)	0.835	0.842 (+0.8%)	+0.4%
$\alpha = 1.4$					
100	0.698	0.707 (+1.3%)	0.687	0.696 (+1.3%)	-0.3%
500	0.841	0.847 (+0.7%)	0.835	0.826 (-1.1%)	-1.8%
1000	0.886	0.892 (+0.7%)	0.882	0.881 (-0.1%)	-0.6%
2000	0.923	0.928 (+0.5%)	0.920	0.924 (+0.4%)	+0.1%
4000	0.951	0.958 (+0.7%)	0.950	0.954 (+0.4%)	+0.3%

in which direct queries could be made to sibling caches or more than one parent cache (arbitrary network configuration).

REFERENCES

[1] K. Zheng, Z. Yang, K. Zhang, P. Chatzimisios, K. Yang, and W. Xiang, "Big data-driven optimization for mobile networks toward 5G," *IEEE Network*, vol. 30, no. 1, pp. 44–51, Jan. 2016.

[2] E. G. Coffman, Jr. and P. J. Denning, *Operating Systems Theory*. Prentice Hall Professional Technical Reference, 1973.

[3] M. Garetto, E. Leonardi, and V. Martina, "A unified approach to the performance analysis of caching systems," *ACM Transactions on Modeling and Performance Evaluation of Computing Systems*, vol. 1, no. 3, pp. 12:1–12:28, May 2016.

[4] M. Garetto, E. Leonardi, and S. Traverso, "Efficient analysis of caching strategies under dynamic content popularity," in *IEEE INFOCOM*, Hong Kong, Apr. 2015, pp. 2263–2271.

[5] E. J. O'Neil, P. E. O'Neil, and G. Weikum, "The LRU-K page replacement algorithm for database disk buffering," *ACM SIGMOD Record*, vol. 22, no. 2, pp. 297–306, Jun. 1993.

[6] T. Johnson and D. Shasha, "2Q: A low overhead high performance buffer management replacement algorithm," in *VLDB*, Santiago de Chile, Chile, Sep. 1994, pp. 439–450.

[7] N. Megiddo and D. S. Modha, "Outperforming LRU with an adaptive replacement cache algorithm," *Computer*, vol. 37, no. 4, pp. 58–65, Apr. 2004.

[8] H. Che, Y. Tung, and Z. Wang, "Hierarchical web caching systems: modeling, design and experimental results," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 7, pp. 1305–1314, Sep. 2006.

[9] V. Martina, M. Garetto, and E. Leonardi, "A unified approach to the performance analysis of caching systems," in *IEEE INFOCOM*, Toronto, Canada, Apr. 2014, pp. 2040–2048.

[10] I. Psaras, R. G. Clegg, R. Landa, W. K. Chai, and G. Pavlou, "Modelling and evaluation of CCN-caching trees," in *IFIP Networking*, Valencia, Spain, May 2011, pp. 78–91.

[11] S. Imai, "Design, modelling, and evaluation of efficient caching mechanisms for content dissemination networks," Ph.D. dissertation, Osaka University, Feb. 2015.

[12] N. Gast and B. V. Houdt, "Asymptotically exact TTL-approximations of the cache replacement algorithms LRU(m) and h-LRU," in *ITC*, vol. 01, Würzburg, Germany, Sep. 2016, pp. 157–165.

[13] A. Dan and D. Towsley, "An approximate analysis of the LRU and FIFO buffer replacement schemes," in *SIGMETRICS*, Boulder, CO, May 1990, pp. 143–152.

[14] C. Fricker, P. Robert, and J. Roberts, "A versatile and accurate approximation for LRU cache performance," in *ITC*, Anaheim, CA, Sep. 2012, pp. 1–8.

[15] C. Yi, A. Afanasyev, I. Moiseenko, L. Wang, B. Zhang, and L. Zhang, "A case for stateful forwarding plane," *Computer Communications*, vol. 36, no. 7, pp. 779 – 791, Apr. 2013.

[16] J. Boyar, M. R. Ehmsen, and K. Larsen, "Theoretical evidence for the superiority of LRU-2 over LRU for the paging problem," in *Approximation and Online Algorithms*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007, vol. 4368, pp. 95–107.

[17] N. Laoutaris, H. Che, and I. Stavrakakis, "The LCD interconnection of LRU caches and its analysis," *Performance Evaluation*, vol. 63, no. 7, pp. 609–634, Jul. 2006.

[18] N. Laoutaris, S. Syntila, and I. Stavrakakis, "Meta algorithms for hierarchical web caches," in *IEEE IPCC*, Phoenix, AZ, Apr. 2004, pp. 445–452.

[19] I. Psaras, W. K. Chai, and G. Pavlou, "Probabilistic in-network caching for Information-Centric Networks," in *ACM SIGCOMM Workshop on Information-Centric Networking*, Helsinki, Finland, Aug. 2012, pp. 55–60. [Online]. Available: http://doi.acm.org/10.1145/2342488.2342501

[20] Z. Ming, M. Xu, and D. Wang, "Age-based cooperative caching in Information-Centric Networking," in *IEEE ICCCN*, Shanghai, China, Aug. 2014, pp. 1–8.

[21] E. J. Rosensweig, J. Kurose, and D. Towsley, "Approximate models for general cache networks," in *IEEE INFOCOM*, San Diego, CA, Mar. 2010, pp. 1–9.

[22] G. Carofiglio, M. Gallo, L. Muscariello, and D. Perino, "Modeling data transfer in Content-centric Networking," in *ITC*, San Francisco, CA, Sep. 2011, pp. 111–118.

[23] A. Dabirmoghaddam, M. Barijough, and J. Garcia-Luna-Aceves, "Understanding optimal caching and opportunistic caching at "the edge" of Information-Centric Networks," in *ACM ICN*, Paris, France, Sep. 2014, pp. 47–56.

[24] 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," in *Sigmetrics*, Austin, TX, Jun. 2014, pp. 595–596.

[25] N. C. Fofack, P. Nain, G. Neglia, and D. Towsley, "Analysis of TTL-based cache networks," in *ICST Performance Evaluation Methodologies and Tools*, Cargese, France, Dec. 2012, pp. 1–10.

[26] R. Chiocchetti, D. Rossi, and G. Rossini, "ccnSim: An highly scalable CCN simulator," in *IEEE ICC*, Budapest, Hungary, Jun. 2013, pp. 2309–2314.

[27] K. Katsaros, G. Xylomenos, and G. C. Polyzos, "Multicache: an overlay architecture for Information-Centric Networking," *Special Issue on*

Architectures and Protocols for the Future Internet Computer Networks, vol. 55, no. 4, pp. 936 – 947, Mar. 2011.

[28] C. Bernardini, T. Silverston, and O. Festor, "A comparison of caching strategies for Content-Centric Networking," in *IEEE GCC*, San Diego, CA, Dec. 2015, pp. 1–6.

[29] D. Rossi and G. Rossini, "On sizing CCN content stores by exploiting topological information," in *IEEE ICCCN*, Orlando, FL, Mar. 2012, pp. 280–285.

Appendix A
Calculation of S_1

Equation (15) is a differential-difference equation. To begin, define $G(x) = F(x + \frac{\tau}{2})$, so that the equation becomes

$$G'(x - \tau/2) = -\lambda_i e^{-\lambda_i(\tau - x)} G(\tau/2 - x). \tag{24}$$

Then with the change of variables $y = x - \tau/2$,

$$G'(y) = -\lambda_i e^{-\lambda_i(\tau/2 - y)} G(-y). \tag{25}$$

Unfortunately there is no standard formula for solving an equation like this. However, with some educated guesses and trial and error, we are able to obtain a solution.

Any function of the form ae^{by} (for constants a and b) cannot be a solution. The next step is to try a function of the form $a_1 e^{b_1 y} + a_2 e^{b_2 y}$ for constants $a_{1,2}$ and $b_{1,2}$. We guess that b_1 and b_2 are two roots of a quadratic, and take the form $b_{1,2} = \gamma \pm \beta$ for constants γ and β. Because there is only a first derivative on the left side, simply substituting $a_1 e^{b_1 y} + a_2 e^{b_2 y}$ will give a linear equation in b_1 and b_2. One way to get a quadratic in $b_{1,2}$ is to set $a_1 = \frac{1}{\sqrt{b_1}}$ and $a_2 = \pm\frac{1}{\sqrt{b_2}}$. So (after some trial and error), we settle on the general solution

$$G(y) = \frac{1}{\sqrt{\gamma + \beta}} e^{(\gamma+\beta)y} \pm \frac{1}{\sqrt{\gamma - \beta}} e^{(\gamma-\beta)y}. \tag{26}$$

We try the '$-$' solution first. Substituting into 25,

$$\begin{aligned} 0 = &\frac{e^{(\lambda_i + 2\gamma)y - \lambda_i \frac{\tau}{2}}}{\sqrt{\gamma^2 - \beta^2}} \\ &\times \left(\sqrt{\gamma + \beta} e^{(\gamma+\beta)y} - \sqrt{\gamma - \beta} e^{(\gamma-\beta)y} \right) \\ &\times \left(\sqrt{\gamma^2 + \beta^2} e^{(2\gamma - \lambda_i)y + \lambda_i \frac{\tau}{2}} - \lambda_i \right). \end{aligned} \tag{27}$$

The first term cannot be 0. The middle factor has no dependence on λ_i, so we move to the third factor. γ and β must be independent of y, and this can be achieved by setting $\gamma = \lambda_i/2$. The third factor then reduces to

$$\sqrt{(\lambda_i/2)^2 - \beta^2} e^{\lambda_i \tau/2} - \lambda_i, \tag{28}$$

which has roots

$$\beta = \pm \frac{\lambda_i}{2} \sqrt{1 - 4e^{-\lambda_i \tau}}. \tag{29}$$

It does not matter which one we pick, so take the '$+$' root. Note that if we had taken the '$+$' solution in (26), then we would not have been able to solve the resulting equation, so we made the right choice. The general solution is

$$G(y) = \frac{1}{\sqrt{\mu_1}} e^{\mu_1 y} - \frac{1}{\sqrt{\mu_2}} e^{\mu_2 y}, \tag{30}$$

with

$$\mu_{1,2} = \frac{\lambda_i}{2}(1 \pm \sqrt{1 - 4e^{-\lambda_i \tau}}). \tag{31}$$

Thus

$$F(x) = G(x - \tau/2) = \frac{1}{\sqrt{\mu_1}} e^{\mu_1 (x - \frac{\tau}{2})} - \frac{1}{\sqrt{\mu_2}} e^{\mu_2 (x - \frac{\tau}{2})}. \tag{32}$$

This determines the solution to (15), up to a constant factor. The solution we want is $\mathbb{E}[S_1(x)] = RF(x)$ for some constant R. We determine the appropriate value of R for our situation via the boundary condition $\mathbb{E}[S_1(\tau)] = \frac{1}{\lambda_i}$, implying

$$R = \frac{1}{\lambda_i} \left(\frac{1}{\sqrt{\mu_1}} e^{\mu_1 \frac{\tau}{2}} - \frac{1}{\sqrt{\mu_2}} e^{\mu_2 \frac{\tau}{2}} \right)^{-1}. \tag{33}$$

Appendix B
Calculation of S_2

For now we assume that $y < \tau$. Equation (21) is almost the same as (15), the only difference being the lack of a minus sign on the right side. Unsurprisingly, the same ideas as before all work. The only difference is at (26), where we take the '$+$' solution instead of the '$-$' one. The general solution is

$$H(y) = \frac{1}{\sqrt{\mu_1}} e^{\mu_1 (y - \frac{\tau}{2})} + \frac{1}{\sqrt{\mu_2}} e^{\mu_2 (y - \frac{\tau}{2})}. \tag{34}$$

This is the solution to (21), up to a constant factor, so that $\mathbb{E}[S_2(y)] = TH(y)$ for some T. Taking $y \to \tau$ gives $T = \frac{\mathbb{E}[S_2(\tau)]}{H(\tau)}$.

On the other hand, taking $y \to \tau$ in (20) gives

$$\mathbb{E}[S_2(\tau)] = \frac{1}{\lambda_i} + \int_0^\infty \lambda_i e^{-\lambda_i u} \mathbb{E}[S(u)] du. \tag{35}$$

Consider the case that $y > \tau$. It is impossible to terminate after u_1'; we are forced to sample (at least) u_2'. Thus the time at which we terminate is independent of y, and so too is $S_2(y)$. Thus $\mathbb{E}[S_2(y)]$ is a constant, and we have $\mathbb{E}[S_2(y)] = \mathbb{E}[S_2(\tau)]$ for $y \geq \tau$. Splitting the integral of (35) into two parts,

$$\begin{aligned} \mathbb{E}[S_2(\tau)] = &\int_0^\tau \lambda_i e^{-\lambda_i u} \mathbb{E}[S_2(u)] du \\ &+ \int_\tau^\infty \lambda_i e^{-\lambda_i u} \mathbb{E}[S_2(u)] du + \frac{1}{\lambda_i} \\ = &\int_0^\tau \lambda_i e^{-\lambda_i u} \frac{\mathbb{E}[S_2(\tau)]}{H(\tau)} H(u) du \\ &+ \int_\tau^\infty \lambda_i e^{-\lambda_i u} \mathbb{E}[S_2(\tau)] du + \frac{1}{\lambda_i} \\ = &\frac{1}{\lambda_i} + Q\mathbb{E}[S_2(\tau)] + e^{-\lambda_i \tau} \mathbb{E}[S_2(\tau)] \end{aligned} \tag{36}$$

where

$$\begin{aligned} Q = &\int_0^\tau \lambda_i e^{-\lambda_i u} \frac{H(u)}{H(\tau)} du \\ = &\frac{2}{\sqrt{\mu_1} e^{\mu_2 \frac{\tau}{2}} + \sqrt{\mu_2} e^{\mu_1 \frac{\tau}{2}}} \\ &\times \left(\sqrt{\mu_1} \sinh\left(\frac{\mu_2 \tau}{2}\right) + \sqrt{\mu_2} \sinh\left(\frac{\mu_1 \tau}{2}\right) \right). \end{aligned} \tag{37}$$

Simplifying (36) results in (22).