Stationary Distribution of a Generalized LRU-MRU Content Cache

George Kesidis
School of EECS, Pennsylvania State University, University Park, PA, USA

Abstract

Many different caching mechanisms have been previously proposed [1], [2], exploring different insertion and eviction policies and their performance individually and as part of caching networks. We obtain a novel closed-form stationary invariant distribution for a generalization of LRU and MRU caching nodes under a reference Markov model. Numerical comparisons are made with an “Incremental Rank Progress” (IRP a.k.a. CLIMB) and random eviction (a.k.a. random replacement) methods under a steady-state Zipf popularity distribution. The range of cache hit probabilities is smaller under MRU and larger under IRP compared to LRU. We conclude with the invariant distribution for a special case of a random-eviction caching tree-network and associated discussion.

I. INTRODUCTION

Caching is a ubiquitous mechanism in communication and computer systems. The role of a content caching network is to reduce the load on the origin servers of requested objects, reduce the required network bandwidth to transmit content\(^1\), and reduce the response times to the queries, cf. Figure 4. Caching in computational settings reduces delays associated with disk IO (page caches). Data actively being, or likely soon to be, accessed by a CPU is stored in lower-level caches, i.e., memories closer (with less access time) to the CPU.

The invariant distribution of the widely deployed Least Recently Used (LRU) eviction mechanism for a caching node was found in [15]. LRU has lower average miss rate compared to FIFO caching\(^2\) [3], [1], [4]. Numerically useful approximations for LRU are found in [10], [9]. The approximation of [9] was clarified in [12]. In [14], LRU caching was studied for dependent (semi-Markov) object demand processes in a limiting regime for certain object popularity profiles. In [5], [7], time-to-live (TTL) caching networks are studied. Approximations for networks of “capacity driven” caches are studied in [16], [13].

Under Most Recently Used (MRU) eviction, the youngest item in the cache is evicted upon cache miss. MRU is used in cases where the older the item is in the cache, the more likely it is to be accessed [11]; or, typically youngest items in the cache are not likely to be needed again soon after they are queried for. In terms of steady-state (long-term) average popularity, MRU may be used in cases where the majority of the aggregate queries are for a set of objects that are individually unpopular (cold), while the set of individually popular (hot) items collectively are a minority of the aggregate queries. If the cache is sufficiently large to hold them all, the individually most popular items are typically the oldest ones in the MRU cache in steady state, and hence are least likely to be evicted by the churn among the younger, unpopular ones.

In this paper, we focus on single caching nodes and present a closed-form invariant distribution for a standard Markov model of a generalization of LRU and MRU eviction. To this end, we provide a proof which will subsequently adapt. For a Zipf popularity distribution, numerical comparisons are made with the simple Incremental Rank Progress (IRP)\(^3\) and and Random Eviction (Random Replacement [1]) methods. Our numerical examples focus on the range of cache hit probabilities for steady-state Zipf popularity distributions. We numerically show that the range of cache hit probabilities is smaller under MRU and larger under IRP compared to LRU, and conjecture that this is true in general. We conclude with a discussion of RE caching networks to give a result for a special case and show why capacity-driven caching networks are difficult to analyze.

---

\(^{1}\)That is, content that is not encrypted for particular end-users.

\(^{2}\)Under FIFO caching, the oldest item in the cache is evicted upon a cache miss.

\(^{3}\)Called CLIMB in [1], IRP is somewhat related to the insertion scheme based on tandem virtual caches of “\(k\)-LRU” [13].
II. BACKGROUND

A. Markov model of Least Recently Used (LRU) eviction policy

The stationary state-space $\mathcal{R}$ of a LRU cache is the set of $B$-permutations of $\{1, 2, \ldots, N\}$ where $N$ is the number of objects that could be cached and $B$ is the capacity of the cache with $N > B > 0$ (typically $N \gg B$) and the objects assumed identically sized (but cf., (6)). For $r \in \mathcal{R}$, define $r(k)$ as the element of $r$ in the $k^{th}$ position. The entries of $r$ are ranked in order of their position in $r$:

- the most recently accessed (LRU) object being $r(1)$,
- the oldest object in the cache being $r(B)$, and
- uncached objects $n$ are denoted $n \notin r$.

Note that in a transient regime, the cache may be in a state $\notin \mathcal{R}$ with fewer than $B$ objects cached.

For a single node, we assume that demand process for object $n \in \{1, 2, \ldots, N\}$ is Poisson with intensity $\lambda_n$. The Poisson demands are assumed independent. Let the total demand intensity be $\Lambda = \sum_{n=1}^{N} \lambda_n$. So, this is the classical “Independent Reference Model” (IRM) with query probabilities $p_n = \lambda_n/\Lambda$ [1], [4].

For LRU, a cache miss of object $r(1)$ at state $M_n^{-1}(r)$ resulting in a transition to state $r \in \mathcal{R}$ occurs at rate $\lambda_{r(1)}$, where $n \notin r$ and

$$ (M_n^{-1}(r))(k) = \begin{cases} n & \text{if } k = B \\ r(k+1) & \text{if } k < B \end{cases} $$

i.e., $n \notin r$ is the oldest object in the cache in state $M_n^{-1}(r)$.

For LRU, a cache hit of object $r(1)$ at state $H_k^{-1}(r)$ resulting in a transition to state $r \in \mathcal{R}$ occurs at rate $\lambda_{r(1)}$ where $1 \leq k \leq B$ and

$$ (H_k^{-1}(r))(\ell) = \begin{cases} r(1) & \text{if } \ell = k \\ r(\ell+1) & \text{if } \ell < k \\ r(\ell) & \text{if } k < \ell \leq B \end{cases} $$

i.e., $r(1)$ is the $k^{th}$ youngest object in the cache in state $H_k^{-1}(r)$ and $H_1^{-1}(r) = r$.

As commonly assumed with the IRM [16], we also assume (i) that cache misses cause the query to be forwarded, possibly to a server holding the requested object, and once resolved, the object is reverse-path forwarded so that caches that missed it can be updated; and (ii) the required time for this query resolution process is negligible compared to the inter-querying times of the caching network.

B. LRU stationary invariant distribution

The following invariant of LRU found by W.F. King in [15].

**Theorem 2.1:** The unique invariant distribution of the LRU Markov chain is

$$ \pi(r) = \frac{1}{\eta} \prod_{k=1}^{B} \frac{\lambda_{r(k)}}{\lambda - \sum_{i=1}^{k-1} \lambda_{r(i)}} \tag{1} $$

for $r \in \mathcal{R}$, where $\forall k$, $\sum_{i=k}^{k-1} \prod_{\eta \in \mathcal{R}} \equiv 0$ [15].

**Proof:** The full balance equations are: $\forall r \in \mathcal{R}$,

$$ (\lambda - \lambda_{r(1)}) \pi(r) = \sum_{n \notin r} \lambda_{r(1)} \pi(M_n^{-1}(r)) + \sum_{j=2}^{B} \lambda_{r(1)} \pi(H_j^{-1}(r)). \tag{2} $$

Under (1), for all $n \notin r$,

$$ \pi(M_n^{-1}(r)) = \frac{\lambda_n}{\lambda - \sum_{i=2}^{B} \lambda_{r(i)}} \prod_{k=2}^{B} \frac{\lambda_{r(k)}}{\lambda - \sum_{i=1}^{k-1} \lambda_{r(i)}} $$
Also under (1), for all $j \in \{2, 3, ..., B\}$,
\[
\pi(H^{-1}_j(r)) = \prod_{k=2}^{j} \frac{\lambda_r(k)}{\Lambda - \sum_{i=2}^{k-1} \lambda_r(i)} \cdot \frac{\lambda_r(1)}{\Lambda - \sum_{i=2}^{j} \lambda_r(i)} \cdot \frac{\lambda_r(k)}{\Lambda - \sum_{i=1}^{k-1} \lambda_r(i)}
\]

Substituting into (2) and after some term cancellation, we see that (1) satisfies (2) if and only if
\[
1 = B \prod_{k=3}^{B+1} \frac{\Lambda - \sum_{i=1}^{k-1} \lambda_r(i)}{\Lambda - \sum_{i=2}^{k-1} \lambda_r(i)} + \sum_{j=2}^{B} \prod_{k=3}^{j} \frac{\Lambda - \sum_{i=1}^{k-1} \lambda_r(i)}{\Lambda - \sum_{i=2}^{k-1} \lambda_r(i)} \cdot \frac{\lambda_r(1)}{\Lambda - \sum_{i=2}^{j} \lambda_r(i)}
\]

where $\prod_{k=3}^{2}(...) = 1$.

Regarding (3), consider the following sequence of independent random experiments to fill the cache. Suppose we’re given initially that the first cache entry is $r(2)$. Now sequentially, according to the distribution (1), object $r(1)$ attempts to enter the cache after $r(2)$. If it fails to enter in the $k$th attempt, then $r(k+2)$ is placed in the cache instead and $r(1)$ tries again. The summand of (3) with $j = 2$ is the probability that $r(1)$ enters in the second position right after $r(2)$: $\lambda_r(1)/(\Lambda - \lambda_r(2))$. Generally, the summand for $j \in \{2, 3, ..., B\}$ is the probability $r(1)$ enters in the $j$th position (after having failed to enter in one of the more highly ranked ones). The first term of the right-hand-side of (3) is the probability $r(1)$ fails to enter the cache. So, (3) must generally hold by the law of total probability.

Finally, since the stationary LRU Markov chain is irreducible on $\mathcal{R}$, there is a unique invariant.

This result was generalized in in [17] to add object-dependent insertion probabilities interpreted as access costs. Also note that, generally, the LRU Markov chain is neither time-reversible nor quasi-reversible [8]. Obviously, more popular objects (larger $\lambda$) are more likely stored, and the LRU invariant is uniform in the special case that all the mean querying rates $\lambda_n$ are the same. Finally, by PASTA, the stationary hit probability of object $n$ in a LRU cache is
\[
h_n = \sum_{r : n \in r} \pi(r),
\]
where the approximations of hit probabilities in [9], [12] are substantially simpler to compute.

C. Incremental Rank Progress (IRP or “CLIMB” [1]) upon query

Under LRU, a query for any object $n$ results in it being ranked first in the cache. One can also consider slowing the “progress through the ranks” of objects as they are queried, leading to some obvious trade-offs with LRU: slowing progress would mean less popular content does not enter the cache at first rank, but also more popular content will take longer to reach the cache. Such issues are important when there are dynamic changes/churn in objects cached and their popularity.

Under an Incremental Rank Progress (IRP) caching mechanism, a query for object $n$ results in its rank improved by just one (or zero if the object is already ranked first), i.e., for $1 \leq k \leq B - 1$, $r \in \mathcal{R}$,
\[
(T_k(r))(\ell) = \begin{cases} 
  r(k) & \text{if } \ell = k + 1 \\
  r(k + 1) & \text{if } \ell = k \\
  r(\ell) & \text{else}
\end{cases}
\]
where the transition \( T_k(r) \rightarrow r \) with rate \( \lambda_{r(k)} \). Missed objects enter the cache at lowest rank, i.e., for \( n \notin r \), define

\[
(S_n(r))(\ell) = \begin{cases} 
  r(k) & \text{if } \ell < B \\
  n & \text{if } \ell = B 
\end{cases}
\]

where the transition \( S_n(r) \rightarrow r \) with rate \( \lambda_{r(B)} \). The invariant for IRP (CLIMB) is found in [1] and can be immediately shown using detailed balance.

**Theorem 2.2:** IRP is time-reversible with unique stationary invariant

\[
\pi(r) = \frac{\prod_{k=1}^{B} \lambda^{B+1-k}_{r(k)}}{\sum_{r' \in R} \prod_{k=1}^{B} \lambda^{B+1-k}_{r'(k)}}.
\]

**D. Random Eviction (RE or Random Replacement (RR) in [1]) upon cache miss without cache rankings**

Suppose that a cache miss of object \( n \) at state \( M_{B,n}^{-1}(r) \) results in a transition to state \( r \in R \) at rate \( B^{-1}\lambda_n \), where \( n \in r \), \( n \notin M_{B,n}^{-1}(r) \), \( \ell \in M_{B,n}^{-1}(r) \), and \( \ell \notin r \). That is, a cache miss for object \( n \) results in \( n \) inserted into the cache and evicting of an object \( \ell \) selected uniformly at random from the cache. The cache state \( r \) does not change if a cache hit occurs. The stationary state-space \( R \) is the set of \( B \)-combinations of \( N \) different objects. The following invariant for RE (RR) is also found in [1] and can also be immediately shown by detailed balance.

**Theorem 2.3:** The RE Markov chain is time-reversible with unique stationary invariant distribution

\[
\pi(r) = \frac{\prod_{n \in r} \lambda_n}{\sum_{r' \in R} \prod_{n \in r'} \lambda_n}.
\]

**E. Considering objects with different lengths**

To account for objects of different lengths for a capacity-driven caches (with ranked objects) like LRU, simply consider a “complete-rankings” LRU variation, where the ranking of all objects is maintained whether the objects are cached or not. That is, the state-space \( R \) is now the set of permutations of all \( N \) objects.

**Corollary 2.1:** The unique stationary invariant \( \pi \) of complete-rankings LRU is (1) with \( B \) replaced by \( N \).

Additionally consider the different sizes \( \ell_n \) bytes of objects \( n \), where the cache capacity \( B \) is in bytes. The number of objects in the cache is given by

\[
K(r) = \max\{K \mid \sum_{k=1}^{K} \ell_{r(k)} \leq B, \ 1 \leq K \leq N\}.
\]

So, the hit probability of object \( n \) when the objects are of variable length is

\[
h_n = \sum_{r : r(n) \leq K(r)} \pi(r).
\]

See the byte-hit performance metric of [2].

**III. Most Recently Used (MRU) eviction**

Again define the state-space \( R \) as the set of \( B \)-permutations of \( \{1,2,\ldots,N\} \). Under MRU [11], [2], a cache hit of object \( r(1) \) at state \( H_{B}^{-1}(r) \) resulting in a transition to state \( r \) occurs at rate \( \lambda_{r(1)} \) where \( 1 \leq k \leq B \) and \((H_{B}^{-1}(r))(\ell)\) is given by (1) as LRU. But for MRU, a cache miss of object \( r(1) \) at state \( M_{B,n}^{-1}(r) \) resulting in a transition to state \( r \in R \) occurs at rate \( \lambda_{r(1)} \), where \( n \notin r \) and

\[
(M_{B,n}^{-1}(r))(k) = \begin{cases} 
  n & \text{if } k = 1 \\
  r(k) & \text{if } k > 1 
\end{cases}
\]

i.e., \( n \notin r \) is the youngest object in the cache in state \( M_{B,n}^{-1}(r) \).

**Theorem 3.1:** The unique invariant distribution of the MRU Markov chain is, for \( r \in R \),

\[
\pi(r) = \frac{\lambda_{r(1)}}{\Lambda} \cdot \frac{1}{\binom{N-1}{B-1}} \prod_{k=2}^{B-1} \frac{\lambda_{r(k)}}{\Lambda - \sum_{i=1}^{k-1} \lambda_{r(i)} - \sum_{n \notin r} \lambda_n}.
\]


Proof: The full balance equations are as for LRU but with a different definition for $M_n^{-1}$.

Let $\Lambda_{-r} = \Lambda - \sum_{n \not\in r} \lambda_n$. By substituting (7) into the full balance equations (and moving the cache-miss terms to the left-hand side), we get that (7) satisfies the full balance equations if and only if

$$1 = \frac{1}{\Lambda_{-r} - \lambda_r(1)} \left( \lambda_r(1) \sum_{j=2}^{B-1} \prod_{k=2}^{j-1} \frac{\Lambda_{-r} - \sum_{i=1}^{j-1} \lambda_r(i)}{\Lambda_{-r} - \sum_{i=2}^{j} \lambda_r(i)} \right)$$

$$+ \lambda_r(B) \prod_{k=2}^{B-1} \frac{\Lambda_{-r} - \sum_{i=1}^{k-1} \lambda_r(i)}{\Lambda_{-r} - \sum_{i=2}^{k} \lambda_r(i)}$$

$$= \sum_{j=2}^{B-1} \left( \prod_{k=2}^{j-1} \frac{\Lambda_{-r} - \sum_{i=1}^{k-1} \lambda_r(i)}{\Lambda_{-r} - \sum_{i=2}^{k} \lambda_r(i)} \right) \frac{\lambda_r(1)}{\Lambda_{-r} - \sum_{i=2}^{j} \lambda_r(i)}$$

$$+ \prod_{k=2}^{B-1} \frac{\Lambda_{-r} - \sum_{i=1}^{k-1} \lambda_r(i)}{\Lambda_{-r} - \sum_{i=2}^{k} \lambda_r(i)}$$

(8)

where $\prod_{k=2}^{1}(...) = 1$.

Regarding (8), consider the following sequence of independent random experiments to determine the position of object $\lambda_r(1)$ when filling the cache, given that only objects $\in r$ will be chosen and that $\lambda_r(2)$ has already been chosen first. $\lambda_r(1)$ is chosen on the first try with probability $\lambda_r(1)/\Lambda_{-r}$, otherwise $\lambda_r(3)$ enters the cache - this is the summand of (8) with $j = 2$. Generally, the $j$th summand is the probability that $\lambda_r(1)$ enters the cache on the $(j-1)^{th}$ try, otherwise object $\lambda_r(j+1)$ is placed in the cache. The final term of (8) is the probability $r(1)$ fails to enter the cache before the last $(B^{th})$ position, because in the penultimate choice only objects $r(B)$ and $r(1)$ remain, i.e., $\lambda_r(B) = \Lambda_{-r} - \sum_{i=1}^{B-1} \lambda_r(i)$. So, (3) must generally hold by the law of total probability.

Finally, since the stationary LRU Markov chain is reducible on $\mathcal{R}$, there is a unique invariant.

Note that it’s easily directly verified that (7) satisfies (2) for the cases $B = 2$ and $B = 3$, e.g., for $B = 3$ and $N = 4$,

$$\pi(r) = \lambda_r(1) \lambda_r(2)/(3\Lambda(\lambda_r(2) + \lambda_r(3))).$$

To interpret (7): $\lambda_r(1)$ is chosen with probability $\lambda_r(1)/\Lambda$; then the remaining $B - 1$ objects in $r$ are chosen from the remaining $N - 1$ objects uniformly at random with probability $(\frac{N-1}{B-1})^{-1}$; finally, the order of the remaining items $\lambda_r(2), \lambda_r(3)$ are determined as the LRU invariant distribution (1).

IV. GENERALIZATION OF LRU AND MRU

“$k$ Recently Used” ($k$RU) is a simple generalization of LRU and MRU wherein object $r(k)$, for some fixed $k \in \{1, 2, ..., B\}$, is evicted upon cache miss; otherwise cache insertion (at rank 1) upon misses and promotion (to rank 1) and demotions (by 1) upon hits are the same as both MRU and LRU. That is, $k$RU is LRU and 1RU is MRU.

Corollary 4.1: The invariant distribution of $k$RU is

$$\pi(r) = \prod_{j=1}^{k} \frac{\lambda_r(j)}{\Lambda - \sum_{i=2}^{j} \lambda_r(i)} \times \frac{1}{\binom{N-k}{B-k}} \prod_{j=k+1}^{B-1} \frac{\lambda_r(j)}{\Lambda - \sum_{i=1}^{j-1} \lambda_r(i) - \sum_{n \not\in r} \lambda_n}. \quad (9)$$
V. NUMERICAL RESULTS FOR SMALL $N, B$

In this numerical study, we directly computed the invariants $\pi$ by generating all possible object permutations representing cache state by the Steinhaus-Johnson-Trotter algorithm. So, we considered only small values for the number of objects and the cache size. Figure 1 is representative of our numerical study on cache-hit probabilities (note that $\sum_{n=1}^{N} h_n = B$) using a Zipf popularity model $\lambda_n = n^{-\alpha}$ for with $\alpha = 0.75$ (see Table 1 of [6]) and most popular object indexed 1 with normalized rate $\lambda_1 = 1$.

$k$RU with $1 < k < B$ gives hit-probability performance between MRU ($k = 1$) and LRU ($k = B$). That is, one can see that the range of hit probabilities for LRU is larger than that of MRU.

![Fig. 1. $k$RU cache hit probabilities $h_n$ and popularity $\lambda_n$ versus object index $n$ for a cache of size $B = 6$, $N = 12$ objects, and Zipf popularity parameter $\alpha = 0.75$, where LRU= 6RU and MRU=1RU](image1)

Figure 2 shows the results of a typical simulation study of $k$RP with cache entry at lowest rank $B$ upon cache miss compared to LRU. Note that $k$RP has greater range of hit probability values than LRU. We postulate that generally for Zipf popularity distributions, the range of hit probabilities of IRP is larger than those of LRU which are larger than those of MRU.

![Fig. 2. $k$RP (with cache entry upon cache miss) and LRU cache hit probabilities $h_n$ and popularity $\lambda_n$ versus object index $n$ for a cache of size $B = 6$, $N = 12$ objects, and Zipf popularity parameter $\alpha = 0.75$.](image2)

For the example of Figure 3, RE has a range of hit probabilities between MRU and LRU.

VI. DISCUSSION: NETWORKS OF RE CACHES

The performance of networks of such capacity-driven caches are approximated in e.g., [16], [13]. To illustrate the difficulties with capacity-driven caching networks, now consider the simplest ones based on RE. Though RE caches are time-reversible, a tree of independent local caches whose collective query-misses are forwarded to an Internet cache (also running RE, see Figure 4), is not time-reversible and not operating under the IRM. To see why it’s not time-reversible, consider a cache miss of object $n$ of local cache $q$ of size $B_q$ in state $r_q$, so that object $n_q$ is evicted, and suppose it’s also a miss on the Internet cache of size $b$ in state $R$, so that object $n$ is evicted; this can be reversed with one query (so that states $r_q$ and $R$ are restored) only if $n = n_q$. 
The following result is for the very special case that the Internet cache holds only one object.

**Proposition 6.1:** The invariant distribution \( \pi \) of the network Figure 4 with RE caching and \( b = 1 \) satisfies

\[
\pi(R | \mathcal{X}) = \frac{\sum_q 1\{R \in r_q\} \Lambda_{q, r_q} / B_q}{\sum_q \Lambda_{q, r_q}}
\]

where

\[
\Lambda_{q, r} = \sum_{\ell \not\in x} \lambda_{q, \ell},
\]

\( \sum_{\emptyset} (...) \equiv 0 \), and indicator \( 1_X = 1 \) if \( X \) is true otherwise \( = 0 \).

**Remarks:** \( \pi(R, r) = \pi(R | \mathcal{X}) \pi(\mathcal{X}) = \pi(R | \mathcal{X}) \prod_q \pi(r_q) \) where \( \pi(r_q) \) is given by (5). In steady state, \( R \subset \cup_q r_q \) a.s., i.e., if \( \forall q, R \not\in r_q \) then \( \pi(R | \mathcal{X}) = 0 \). Also, \( \Lambda_{q, r_q} \) is the rate of cache misses at local cache \( q \) when it’s in state \( r_q \).

One can identify \( \hat{\lambda}_n := \sum_q \lambda_{q, n} (1 - h_{q, n}) \) as the incident mean rate of queries for object \( n \) to the Internet cache, where \( 1 - h_{q, n} \) is the stationary miss probability of local cache \( q \) for object \( n \), \( \pi(\mathcal{X}) = \prod_q \pi(r_q) \) and \( \pi(r_q) \) is given by (5). According to this proposition, \( \pi(R | \mathcal{X}) \) does not depend on the \( \hat{\lambda}_n \), the way the IRM invariant \( \pi(r_q) \) depends on the \( \lambda_{q, n} \) in (5). Note that an individual RE cache \( r \) is not quasi-reversible since the miss rates ("departures"), \( \frac{1}{\pi(r)} \sum_{m \not\in r, n \in r} \lambda_{n} \pi(\delta_{n+m} r) \) depend on the state \( r \). Though quasi-reversibility is not a necessary condition [8], Proposition 6.1 shows that RE networks generally do not have product-form invariants.

**Proof:** For \( n \in r_q, m \not\in r_q \), let \( \delta_{q-n+m} r \) be \( r \) but with \( n \) in \( r_q \) replaced by \( m \). Similarly define \( \delta_{-n+\ell} R \). The
full balance equations are
\[
\pi(r, R) \sum_{q,m: m \not\in r_q} \lambda_{q,m} = \sum_{q,n: n \not\in r_q} \pi(\delta_{q-n+m} r, R) \frac{\lambda_{q,n}}{B_q} + \sum_{q,n, l: m \not\in r_q, n \in r_q \cap R} \pi(\delta_{q-n+m} r, \delta_{-n+l} R) \frac{\lambda_{q,n}}{B_q}\]

Dividing by \(\pi(r, R) = \pi(R|r) \prod_q \pi(r_q)\) and then substituting the stationary joint distribution of the independent RE local caches (5) into the full balance equations gives: \(\forall r, R\),
\[
\pi(R| r) \sum_{q,m: m \not\in r_q} \lambda_{q,m} = \sum_{q,m: m \not\in r_q} \frac{\lambda_{q,m}}{B_q} \times \sum_{n \in r_q \cap R} \left( \pi(R|\delta_{q-n+m} r) + \frac{1}{b} \sum_{l \not\in R} \pi(\delta_{-l-n} R|\delta_{q-n+m} r) \right)
\]

For the special case of \(b = 1\), i.e., \(R = n\) is a single object, we get that the right-hand-side simplifies to
\[
\sum_{q,m: m \not\in r_q} \frac{\lambda_{q,m}}{B_q} \{R \in r_q\} \left( \pi(R|\delta_{q-n+m} r) + \sum_{l \not\in R} \pi(\ell|\delta_{q-n+m} r) \right) = \sum_{q} \frac{\lambda_{q,n}}{B_q} \{R \in r_q\}
\]
The invariant is unique since \((r, R)\) is irreducible.

For general caching networks \(r\) where \(\phi_{i,j}\) is the probability that cache \(i\) forwards a miss to cache \(j\), we can write the total stationary incident mean querying rate for object \(n\) at cache \(j\) as \(\hat{\lambda}_{j,n} := \lambda_{j,n} + \sum_{i \neq j} \hat{\lambda}_{i,n}(1 - h_{i,n})\phi_{i,j}\) [16], where the stationary hit probabilities \(h_{i,n} = 1\) for origin server \(i\) storing object \(n\). Approximations for stationary LRU caching networks are studied in [16], [13]. Generally, invariants for LRU and other capacity-driven caching networks are not known.

## References

1. O.I. Aven, E.G. Coffman, and Y.A. Kogan. *Stochastic analysis of computer storage*. D. Reidel Publishing Co., 1987.
2. A. Balamash and M. Krunz. An overview of web caching replacement algorithms. *IEEE Communications Surveys & Tutorials*, 6(2), 2004.
3. L.A. Belady, R.A. Nelson, and G.S. Shedler. An anomaly in Space-time Characteristics of Certain Programs Running in a Paging Machine. *Commun. ACM*, 12(6), June 1969.
4. J. Van Den Berg and A. Gandolphi. LRU is better than FIFO under the independent reference model. *J. Appl. Prob.*, 29, 1992.
5. D.S. Berger, S. Singla School, P. Gland, and F. Ciucu. Exact Analysis of TTL Cache Networks The Case of Caching Policies Driven by Stopping Times. In *Proc. ACM SIGMETRICS*, Austin, Texas, June 2014.
6. L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker. Web Caching and Zipf-like Distributions: Evidence and Implications. In *Proc. IEEE INFOCOM*, 1999.
7. F. Cavallin, A. Marin, and S. Rossi. A product-form model for the analysis of systems with aging objects. In *Proc. IEEE MASCOTS*, Atlanta, Sept. 2015.
8. X. Chao, M. Miyazawa, R.F. Serfozo, and H. Takada. Markov network processes with product form stationary distributions. *Queueing Systems*, 28:377401, 1998.
9. H. Che, Y. Tung, and Z. Wang. Hierarchical Web Caching Systems: Modeling, Design and Experimental Results. *IEEE JSAC*, 20(7), Sept. 2002.
10. A. Dan and D. Towsley. An approximate analysis of the LRU and FIFO buffer replacement schemes. *SIGMETRICS Perform. Eval. Rev.*, 18:143 152, April 1990.
[11] S. Dar, M.J. Franklin, B.T. Jonsson, D. Srivastava, and M. Tan. Semantic data caching and replacement. In Proc. Conf. on Very Large Databases (VLDB), 1996.
[12] C. Fricker, P. Robert, and J. Roberts. A Versatile and Accurate Approximation for LRU Cache Performance. In Proc. International Teletraffic Congress, 2012.
[13] M. Garetto, E. Leonardi, and V. Martina. A Unified Approach to the Performance Analysis of Caching Systems. ACM TOMPECs, 1(3), May 2016.
[14] P.R. Jelenkovic and A. Radovanovic. Least-recently-used caching with dependent requests. Theoretical Computer Science, 326:293–327, Oct. 2004.
[15] W.F. King. Analysis of paging algorithms. In Proc. IFIP Congress, Lyublyana, Yugoslavia, Aug. 1971.
[16] E. Rosensweig, J. Kurose, and D. Towsley. Approximate models for general cache networks. In Proc. IEEE INFOCOM, March 2010.
[17] D. Starobinski and D. Tse. Probabilistic methods for web caching. Performance Evaluation, 2001.