Asymptotic analysis methods for list-based caches with non-uniform access costs

Giuliano Casale, Nicolas Gast

***Abstract*—List-based caches can offer lower miss rates than single-list caches, but their analysis is challenging due to state-space explosion. In this setting, we analyze asymptotic performance for a general class of list-based caches with tree structure, non-uniform access costs to items and lists, and random replacement (RR) or first-in first-out (FIFO) replacement policies. After determining a product-form solution for the underlying Markov processes, we obtain analytical expressions for cache performance metrics by means of a singular perturbation method. Simulations and comparison with a refined mean-field approximation indicate that our asymptotic methods rapidly converge to the equilibrium distribution as the number of items and the cache capacity grow in a fixed ratio, with mean average errors typically below 2%. Our methods also enable the analysis of models with synchronous requests and closed queueing networks with caching nodes, extending the applicability of approximations for list-based caches.**

I. Introduction

Replacement policies provide a strategy to select items to replace in a cache and are an important performance factor influencing the design of web systems [1], content delivery networks [2], edge networks [3], and peer-to-peer traffic [4]. We here focus on the theoretical analysis of randomized policies operating within a single cache, a problem that has attracted much attention over the years [5]–[13]. We consider *list-based* caches, which can deliver lower miss rates than single-list caches [14], [15], but due to state-space explosion issues are still not fully understood from a theoretical standpoint. We here focus in particular on a general class of list-based caches that also admits a tree structure for the lists, rather than linear as in earlier work on these models [15], and where access costs to items and lists are probabilistic and in general non-uniform.

This paper stems from the recent development in [15] of the first general framework to analyze *list-based* caches with the random replacement (RR), FIFO, or CLIMB policies, under the independent reference model (IRM) [16], [17]. List-based caches can deliver lower miss rates than single-list caches, even when the latter are equipped with the LRU policy. The present work expands the scope of the theory to tree structured lists, where an item can reach a list according to some probabilistic access costs. Our model is fairly general, as costs can be further differentiated across multiple request streams and depend on the requested item and the list that contains it. Differentiating the item request process into multiple streams is helpful in some studies, such as upon evaluating fairness of space sharing across multiple cache owners [18]. We also consider a further departure from the IRM model by allowing arrival rates to depend on the cache state and to be either asynchronous or synchronous, with the latter imposing a maximum number of outstanding requests for each stream.

Our main contributions to the performance analysis of these caches are as follows. First, we define a Markov process for the list-based RR and FIFO policies with tree structured lists and non-uniform costs. We find that, similarly to the uniform cost case, this process admits a product-form solution. Next, we develop an asymptotic expression of the normalizing constant of state probabilities that allows us to derive the limiting values of the miss rates as the scale of the system grows. To the best of our knowledge, our approach is the first one to apply the singular perturbation method used in geometric optics and queueing theory to the analysis of caches [19]. A refined mean-field approximation is also developed to analyze the asymptotic behavior of the cache in transient and equilibrium regimes. These techniques can inexpensively approximate a cache performance in problems where speed of analysis is critical, such as within numerical optimization programs. We use examples and simulations to illustrate the validity and accuracy of our approximations, both in models where caches are considered in isolation or operate within a network of queueing stations. We also show that our method is competitive against refined mean-field approximations for the models of interest. The latter case is of interest in applications where cache latency needs to be explicitly quantified.

The paper is organized as follows. Section I-1 overviews related work. The reference model is introduced in Section II and its equilibrium analyzed in Section III. An exact analysis is developed in Section IV. Section V approximates the normalizing constant. Iterative analysis and bounds are introduced in Section VI. Numerical results are reported in Section VII. Decomposition approximations for models with synchronous requests and queueing are presented in Section VIII and Section IX, respectively, and are followed by the conclusion. Proofs are given in the Appendix. Additional material such as examples and additional results are included within the supplemental material.

1) Related work: Recently, most of the research literature has focused on time-to-live (TTL) caches, which can be analyzed either by exact methods [20] or by characteristic time approximation [21]. The later approach has also been shown to be valid for randomized policies [22]. However, there is still a limited body of work on using this method to study list-based caches [23] and therefore our paper strikes a

G. Casale is with the Department of Computing at Imperial College London, UK; e-mail: (g.casale@imperial.ac.uk).

N. Gast is with Inria Grenoble, France; e-mail: (nicolas.gast@inria.fr).

The authors would like to thank Benny van Houdt for helpful comments during preparation of this manuscript. This work was supported by the EU H2020 grants DICE (644869) and RADON (825040) and by a UK EPSRC grant (EP/L00738X/1, iBids). Research data is available at http://doi.org/10.5281/zenodo.1134886 (CC BY 4.0 license).

contribution towards performance analysis of this particular family of stochastic systems.

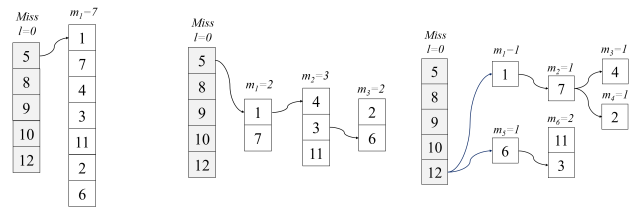
This paper is related to works that consider the independent reference model (IRM), which include [17], [7], [5], [6], [24], and references therein. Although real systems deviate from the IRM assumptions, IRM theoretical results provide useful insights, such as the characterization of the miss ratio of LRU [9], and formulas for the approximate analysis of cache networks [25]. The work in [26] further adopts an IRM model to help designers compare trade-offs in hybrid memory systems, first extending the linear model in [15] to a parallel structure. Recently, some frameworks that simplify the mathematical analysis of caches have been investigated [14], [15]. This paper generalizes some of the ideas in [15], which provides a mathematical framework to study list-based caches. [26] offers another example of generalization of [15], but to list-based caches with flat and layered list topologies. Compared to the above works, our results apply to general tree-structured caches and to non-uniform costs.

Costs are useful in cache modeling to abstract factors such as item importance, fetch latency, optional eviction, or access price [1], [27]–[29]. Models that ignore these factors are said to assume uniform costs. In the literature, [27] is among the first works to analyze the least-recently used (LRU) and CLIMB policies when an item has an access cost. The authors in [28] study optimal policies for a single-list with non-uniform costs. Following the randomization approach proposed in [27], costs are here modeled as probabilities that items are promoted to a deeper list as a result of a cache hit. This definition is rather flexible, for example it can be extended to take into account item sizes through a notion of density [27]. However, models with non-uniform costs are generally harder to deal with than under uniform costs, since the access cost need not be monotonic through the list hierarchy. Other cost-based algorithms are surveyed in [1]. The present work delivers an extension of a preliminary paper that analyzes non-uniform access costs in a less general family of list-based caches with linear structure [30].

II. Preliminaries

Throughout, we use i, k as item indexes, l, j as list indexes, and v as stream index. Our reference model consists of a cache partitioned into h lists, in general arranged as a tree, each having a capacity of m_l items, $l = 1, \dots, h$. We use the convention that a virtual list with index $l = 0$ contains all the items outside the cache. Diagrams illustrating some possible instances of the list-based caches are given in Figure 1.

We denote the capacity vector by $\boldsymbol{m} = (m_1, \dots, m_h)$ and set $m = \sum_{l=1}^{h} m_l$. The total number of items is n and we require it to exceed the cache capacity ($n > m$). We take the usual assumption that items have identical sizes. The time required to replace an item is assumed to be negligible compared to inter-request time, and thus is treated as instantaneous. The cache serves u independent streams, each issuing requests for item k in list l according to a Poisson process with rate $\lambda_{vk}(l)$, $v = 1, \dots, u$, $k = 1, \dots, n$, $l = 0, \dots, h$. A summary of the main notation used in the paper is given in Appendix H of the supplemental material.



(a) Single list (b) Linear structure (c) Tree structure

Fig. 1. Examples of list-based caches with different internal structure, total capacity $m = 7$, and $n = 12$ items.

Let $c_{vk}(l, j)$, $l \neq j$, denote the probability that item k, currently residing in list $l = 0, \dots, h$, moves to access list $j = 0, \dots, h$ after a hit from stream v. Note that, under this definition:

- $c_{vk}(0, e)$ is the probability that item k enters the cache in list e after a cache miss from stream k.
- $c_{vk}(l, l) = 1 - \sum_{j=1, j \neq l}^{h} c_{vk}(l, j)$ is interpreted as the probability that the item is served and remains in its current list l in the same position.

The definition of the above costs is arbitrary and left to the user. For example, costs may be used to restrict access to certain parts of the cache for certain items or streams.

A. List-based replacement policies

The challenge in designing effective replacement policies is that item popularity is not known beforehand, otherwise one could minimize miss rates by keeping the most frequently accessed items in the cache [27]. The focus is therefore on policies that self-organize item storage so to optimize miss rates. List-based replacement policies under uniform access costs include RR$(\boldsymbol{m})$, FIFO$(\boldsymbol{m})$, CLIMB$(\boldsymbol{m})$, and LRU$(\boldsymbol{m})$ [15], [23]. In these policies, when an item i residing in list $l < h$ is requested, under a linear structure the item is promoted to list $l + 1$, demoting an item chosen according to a given criterion.

Our goal is in particular to investigate RR$(\boldsymbol{m})$ and FIFO$(\boldsymbol{m})$ in the context of non-uniform access costs. Since CLIMB$(\boldsymbol{m})$ can be reduced to a special case of these policies, it is implicitly covered by our analysis. Contrary to RR$(\boldsymbol{m})$ and FIFO$(\boldsymbol{m})$, the Markov process of LRU$(\boldsymbol{m})$ is not known to admit an explicit solution in list-based caches and thus it is analyzed using approximations [23]. For this reason, and similarly to prior work on randomized policies [11], [15], [26], LRU$(\boldsymbol{m})$ falls beyond the scope of our investigation, which stems from an exact product-form result that holds only for RR$(\boldsymbol{m})$ and FIFO$(\boldsymbol{m})$.

The variants of these policies that allow for non-uniform costs are denoted by RR-c$(\boldsymbol{m})$, FIFO-c$(\boldsymbol{m})$, LRU-c$(\boldsymbol{m})$, and CLIMB-c$(\boldsymbol{m})$. The last two variants are developed in [27] for single-list caches. The authors use a randomization approach to model the impact of costs on the cache dynamics. Compared to [27], we also adopt randomization, but focus the analysis on list-based caches.

Based on the above definitions, we generalize RR$(\boldsymbol{m})$, FIFO$(\boldsymbol{m})$, and CLIMB$(\boldsymbol{m})$ as follows:

TABLE I
COMPARISON OF THE MISS RATE SUM S_M IN SOME LINEAR AND TREE STRUCTURED CACHES

				RR-c($\boldsymbol{m}$)		FIFO-c($\boldsymbol{m}$)	
m_1	m_2	m_3	m_4	linear	tree	linear	tree
4	0	0	0	0.986	0.986	0.986	0.986
3	1	0	0	0.989	1.081	0.989	1.081
2	2	0	0	0.993	1.210	0.993	1.210
1	3	0	0	0.998	1.378	0.998	1.378
2	1	1	0	0.994	1.210	0.994	1.210
1	2	1	0	0.999	1.378	0.999	1.378
1	1	2	0	1.000	1.378	1.000	1.378
1	1	1	1	0.999	0.978	0.999	0.978
Minimum S_M:				0.986	0.978	0.986	0.978

RR-c($\boldsymbol{m}$).: A cache miss for item i from stream v with probability $c_{vi}(0,e)$ evicts a random item from list e, replacing it with i. Consider now a cache hit for item i in list $l > 0$. If l is not a leaf in the cache tree structure, with probability $c_{vi}(l,l')$ the hit swaps the position of i with a victim item k occupying a random position in list l'; if list l is a leaf then i item does not change position.

FIFO-c($\boldsymbol{m}$).: This policy is similar to RR-c($\boldsymbol{m}$), but upon a cache hit for i, if l is not a leaf, then the victim item k is taken from position $m_{l'}$ in list l', rather than from a random position. All other items in list l' move back by one position, and item i is placed in position 1 of list l'.

CLIMB-c($\boldsymbol{m}$).: This is the limiting case of FIFO-c($\boldsymbol{m}$) and RR-c($\boldsymbol{m}$) when $\boldsymbol{m} = (1,\ldots,1)$ [8].
Note that in all the above policies, the target list l' is chosen with probability $c_{vi}(l,l')$.

1) Example.: In this example, we illustrate the performance of different access cost structures and replacement policies. Numerical results are shown using the exact solutions developed in Section IV. We consider a family of caches with capacity $m = 4$, $n = 9$ items, and $u = 2$ streams, respectively with Poisson arrival rates $\lambda_{1k}(l) = 1$ and $\lambda_{2k}(l) = 3$ for every item k and list l accessible to the stream. Stream accesses only items $k = 1,\ldots,4$, while stream 2 reads only items $k = 5,\ldots,9$; item references follow a Zipf-like distribution with parameter $\alpha_1 = 1.4$ in stream 1 and $\alpha_2 = 0.6$ in stream 2.

Let S_M be the sum of the average miss rates for each stream. We compare S_M in two cases: (i) caches with a linear structure; (ii) caches with a tree structure. In both cases, all items enter in list 1 and are always promoted upon a hit in a list $l < h$. In the tree cache, a hit in list 1 promotes, with probability proportional to the index of the target list, the item in one of the lists $2,\ldots,h-2$ for stream $v = 1$, and in one of the lists $2,\ldots,h$ for stream $v = 2$. Thus, stream 2 has at its disposal more cache space. Note that this structure imposes non-uniform access costs, since stream 1 cannot access all lists.

Results are shown in Table I. The performance for FIFO-c($\boldsymbol{m}$) and RR-c($\boldsymbol{m}$) is identical. Access costs affect the miss rates S_M, which are minimal for a tree-structured cache with $\boldsymbol{m} = (1,1,1,1)$. In this model the access probability to the lists $h = 2,3,4$ in the tree are respectively $0.22\bar{2}$, $0.33\bar{3}$, and $0.44\bar{4}$. This illustrates the point that a tree structure can deliver better performance than a linear structure. This table also illustrates that increasing the number of lists does not always lead to a better performance. It was shown in [15] that when there is a unique stream of requests, then the miss ratio of a cache with a single list is always higher than the miss ratio of any linear structure with two or more lists. Table I shows that it is not the necessarily the case when there is more than one stream as the linear configuration with the best performance is the one with a single list ($m_1 = 4$, $m_2 = m_3 = m_4 = 0$).

III. ANALYTICAL MODEL

A. Markov process

Define the state vector $\boldsymbol{s} = [s(i,j)]$, such that $s(i,j) \in [1,n]$ is the index of the item in position i of list j, and let $\mathcal{S}$ be the space of cache states that are reachable from the initial state. We take the convention that $m_0 = n - m$ and that $s(i,0)$ indicates the item with the ith largest index among those outside the cache. Let $\pi(\boldsymbol{s})$ be the equilibrium probability of the continuous-time Markov chain (CTMC) modeling the replacement policy.

From now on, we restrict our attention to recurrent states and assume the CTMC to be irreducible. We also require that for each list l there exists an item that can reach it under requests from one or more streams.

Let the *access path* $\mathcal{P}_{vi}(j)$ be a sequence of lists visited by an item i that starts outside the cache until it is promoted to list j under requests from stream v only. Note that $\mathcal{P}_{vi}(j)$ is unique, since we assume caches to have a tree structure. If item i cannot reach list j under requests coming from stream v, we set $\mathcal{P}_{vi}(j) = \emptyset$. For example, in a cache with a linear structure $\mathcal{P}_{vi}(j) = \{(0,e),(e,e+1),\ldots,(j-2,j-1)\}$, where e is the list of entry of the item.

B. Product-form result

We now give a product-form result for $\pi(\boldsymbol{s})$ subject to assumptions on the cache structure.

Theorem 1. *Assume that if $\mathcal{P}_{vi}(j) = \emptyset$ then $\lambda_{vi}(j) = 0$ and otherwise that access paths are stream-independent, i.e., $\mathcal{P}_{vi}(j) = \cup_{v=1}^{u}\mathcal{P}_{vi}(j) = \mathcal{P}_i(j)$. Then, RR-c($\boldsymbol{m}$) and FIFO-c($\boldsymbol{m}$) admit the equilibrium distribution*

$$\pi(\boldsymbol{s}) = \frac{\prod_{j=0}^{h}\prod_{i=1}^{m_j}\gamma_{s(i,j)j}}{E(\boldsymbol{m})} \tag{1}$$

for all recurrent states $\boldsymbol{s} \in \mathcal{S}$, where $E(\boldsymbol{m})$ is a normalizing constant and

$$\gamma_{kj} = \begin{cases} \prod_{(l,l')\in\mathcal{P}_{\|}(|)}\sum_{v=1}^{u}\lambda_{vk}(l)c_{vk}(l,l') & \text{if } j > 0 \\ 1 & \text{if } j = 0 \end{cases}$$

for all $k = 1,\ldots,n$ and $j = 0,\ldots,h$.

The proof is given in Appendix A, together with an illustrative example in Appendix I. Compared to the proof techniques used in prior work [15], [26], which are based on the global balance equations of the underlying Markov process, our proof gives for RR-c($\boldsymbol{m}$) an argument based on Kolmogorov's criterion [31].

The above theorem generalizes prior work on list-based caches with a linear structure, as it states that RR($\boldsymbol{m}$)-c and FIFO($\boldsymbol{m}$)-c admit a product-form also if the cache is tree-structured, where access paths can be item-dependent and stream-dependent, the latter restricted to the value of the access costs $c_{vi}(l,l')$ along the path to item i. Note that if streams access disjoint sets of items, then item-dependence provides a way to introduce a form of stream-dependent paths. For example, all items associated to a stream may be cached in a particular subset of lists, provided that they are not accessed by other streams.

We refer to γ_{ij} as the *cost factor* for item i to access list j. The cost factor differs from the access cost in that it accounts for both the costs and the item request rates involved in item i movement from outside the cache into list j. Note that in uniform cost models with list-independent arrivals and linear structure, the cost factor reduces to $\gamma_{ij} = \lambda_i^j$ in continuous-time, where $\lambda_i = \sum_{v=1}^{u} \lambda_{vi}(0)$ is the total request rate for item i, and to $\gamma_{ij} = p_i^j$ in discrete-time, where p_i is the popularity of item i. This special case matches known expressions, such as those in [15].

From the above product-form result, it is also useful to note that since arrivals are Poisson, if items all enter the cache in the same list e and $c_{vi}(0,e) < 1$, then a model with revised costs $c'_{vi0e} = 1$ and rates $\lambda'_{vi}(0) = c_{vi}(0,e)\lambda_{vi}(0)$ has the same equilibrium distribution as the original model.

C. Performance measures

Let $\pi_{kl}(\boldsymbol{m})$ be the (marginal) probability that item k is in list l, and let $\pi_{k0}(\boldsymbol{m})$ denote the miss ratio for item k. By definition we have $\sum_{l=0}^{h} \pi_{kl}(\boldsymbol{m}) = 1$ and since at all times list l contains exactly m_l items we have

$$\sum_{k=1}^{n} \pi_{kl}(\boldsymbol{m}) = m_l \tag{2}$$

Using the product-form result (1), we can write

$$\pi_{kl}(\boldsymbol{m}) = m_l \gamma_{kl} \frac{E_k(\boldsymbol{m} - 1_l)}{E(\boldsymbol{m})} \tag{3}$$

where 1_l is the unit vector in direction l, $E_k(\boldsymbol{m})$ is the normalizing constant of a model without item k, m_l counts the possible positions of k within list l, and by (1) the remaining terms give the probability of states in which item k resides in list l.

Define now $\pi_k(\boldsymbol{m}) = \sum_{l=1}^{h} \pi_{kl}(\boldsymbol{m})$ as the probability that item k is in the cache. Noting that $E_k(\boldsymbol{m})$ encompasses all and only states where k is outside the cache, we can write the miss ratio for item k as

$$\pi_{k0}(\boldsymbol{m}) = \frac{E_k(\boldsymbol{m})}{E(\boldsymbol{m})} = 1 - \pi_k(\boldsymbol{m}) \tag{4}$$

From the definitions, the miss rate for stream v and item k may be written as

$$M_{vk}(\boldsymbol{m}) = \pi_{k0}(\boldsymbol{m}) \lambda_{vk}(0) c_{vk}(0,e) \tag{5}$$

where e. The miss rate of item k is then $M_k(\boldsymbol{m}) = \sum_v M_{vk}(\boldsymbol{m})$ and the cache miss rate is $M(\boldsymbol{m}) = \sum_{v,k} M_{vk}(\boldsymbol{m})$.

IV. Exact analysis

In this section, we derive recurrence relation to calculate the performance measures. The algorithms we derive are exact and require polynomial time and space in n and m for constant h. Thus, they ease the problem that the state space $\mathcal{S}$ grows exponentially in size with n and m, preventing direct numerical analysis of RR-c($\boldsymbol{m}$) and FIFO-c($\boldsymbol{m}$) by (1). Additional results related to the recursive computation of $E(\boldsymbol{m})$ and miss ratios are given in Appendix L in the supplemental material.

A. Recurrence relation for the normalizing constant

We begin by conditioning on the list containing item k, so that we can recursively express the normalizing constant as

$$E(\boldsymbol{m}) = E_k(\boldsymbol{m}) + \sum_{j=1}^{h} m_j \gamma_{kj} E_k(\boldsymbol{m} - 1_j) \tag{6}$$

for an arbitrary $k = 1, \ldots, n$. Here, m_j accounts for the permutations of the positions of item k within list j. This is a slight generalization of a similar expression obtained in [15] for uniform cost models. The recurrence relation requires the boundary conditions $E(\boldsymbol{0}) = 1$, and $E(\boldsymbol{m}) = 0$ if either $\sum_i m_i > n$ or $m_j < 0$ for any j. Further recurrence relations related to (6) are developed in Appendix (L) in the supplemental material.

Using (6), $E(\boldsymbol{m})$ can be computed in $\mathcal{O}(n \prod_{i=1}^{h}(m_i + 1))$ time and space. Note that (6) can incur floating-point range exceptions due to the rapid growth of $E(\boldsymbol{m})$. Such exceptions can be addressed either by exact arithmetic or by dynamic scaling of the normalizing constant value. The latter leverages that if we scale the cost factors as $\gamma'_{kj} = S\gamma_{kj}$, $E(\boldsymbol{m})$ is scaled by a factor S^m due to (1). The factor S can then be dynamically chosen so to scale $E(\boldsymbol{m})$ and avoid range exceptions in the normalizing constant, using standard methods developed in queueing theory [32].

From the normalizing constant, the miss rate $M(\boldsymbol{m})$ can be obtained as follows. Let ξ_l be defined as

$$\xi_l(\boldsymbol{m}) = m_l \frac{E(\boldsymbol{m} - 1_l)}{E(\boldsymbol{m})} \tag{7}$$

and denote $F_l(\boldsymbol{m}) = m_l / \xi_l(\boldsymbol{m})$. Note that by (6) and by the definition of miss rate, we may write

$$M(\boldsymbol{m}) = \frac{E(\boldsymbol{m} + 1_1)}{E(\boldsymbol{m})} = F_1(\boldsymbol{m} + 1_1) = \frac{m_1 + 1}{\xi_1(\boldsymbol{m} + 1_1)} \tag{8}$$

Thus, computational methods for $\xi_l(\boldsymbol{m})$ are sufficient to determine the miss rate $M(\boldsymbol{m})$.

B. Exact analysis of marginal probabilities

We begin by deriving the following relation for marginal state probabilities

Theorem 2. *The probability of finding item k in list l is*

$$\pi_{kl}(\boldsymbol{m}) = \gamma_{kl} \xi_l(\boldsymbol{m})(1 - \pi_k(\boldsymbol{m} - 1_l)) \tag{9}$$

for all items $k = 1, \ldots, n$ and lists $l = 1, \ldots, j$.

The proof is given in Appendix B. We note that (9) is similar to the arrival theorem for queueing networks [33]. It may be seen as relating the equilibrium distribution of the cache with the conditional distribution of states seen by item k while residing in list l. A similar argument holds also for the arrival theorem, which is sometimes seen as relating the equilibrium distribution of a closed product-form queueing network with the conditional distribution while a job resides at a given station [34].

We are now ready to obtain a recurrence relation for the marginal probabilities. Using (2) and (9), and solving for $\xi_l(\boldsymbol{m})$ we find

$$\xi_l(\boldsymbol{m}) = \frac{m_l}{\sum_{k=1}^{n} \gamma_{kl}(1-\pi_k(\boldsymbol{m}-1_l))} \tag{10}$$

Finally, plugging (10) into (9), and using that $\pi_k(\boldsymbol{m}) = \sum_{l=1}^{h} \pi_{kl}(\boldsymbol{m})$, we get the exact recurrence relation

$$\pi_{il}(\boldsymbol{m}) = \frac{m_l \gamma_{il}(1-\sum_{j=1}^{h} \pi_{ij}(\boldsymbol{m}-1_l))}{\sum_{k=1}^{n} \gamma_{kl}(1-\sum_{j=1}^{h} \pi_{kj}(\boldsymbol{m}-1_l))} \tag{11}$$

for all $i = 1,\dots,n$ and $l = 1,\dots,h$. The recursion has termination conditions $\pi_{il}(\boldsymbol{m}) = 0$ if either $\boldsymbol{m} = \boldsymbol{0}$ or $\min_j m_j < 0$. Solving (11) recursively requires $\mathcal{O}(nh \prod_{i=1}^{h}(m_i+1))$ time and $\mathcal{O}(n \prod_{i=1}^{h}(m_i+1))$ space. Note that from the knowledge of the marginal probabilities, we can compute item miss rates by the expressions in Section III-C.

V. Normalizing constant asymptotics

Exact recurrence relations are too expensive to apply for medium and large sized caches. To address this issue, we obtain an asymptotic limit for $E(\boldsymbol{m})$ that can be used to approximate performance measures under non-uniform costs. The main contribution is to develop an asymptotic expansion for $E(\boldsymbol{m})$ in the limit $n \to \infty$ and $m \to \infty$, with $n/m \sim O(1)$. To this aim, we use the singular perturbation method [19], which reformulates multivariate recurrence relations as partial differential equations (PDEs) that can be explicitly solved.

A. Overview

Before giving the formal derivation of the asymptotics, we sketch the solution paradigm of the singular perturbation method first proposed in [35] for queueing network models. We apply this method to the cache as $n, m \to \infty$ with $n/m \sim O(1)$, and take the following steps:

1) We scale recurrence parameters: $\boldsymbol{x} = \boldsymbol{m}\varepsilon$, $y = n\varepsilon$, for some $\varepsilon \to 0$
2) We plug in the quite general analytic form
$$h(\boldsymbol{x}, y) \sim D(\varepsilon) B(\boldsymbol{x}, y) e^{\phi(\boldsymbol{x},y)/\varepsilon} \tag{12}$$
where $h(\boldsymbol{x}, y)$ corresponds to a scaled $E(\boldsymbol{m})$ and $\phi(\boldsymbol{x}, y)$, $B(\boldsymbol{x}, y)$, and $D(\varepsilon)$ are functions to be determined.
3) We expand in powers of ε the resulting equation up to the $O(1)$ and $O(\varepsilon)$ terms.
4) We obtain from the $O(1)$ and $O(\varepsilon)$ terms two partial differential equations with boundary conditions given by known analytical results for a cache with uniform item popularity, finding through their solution $\phi(\boldsymbol{x}, y)$, $B(\boldsymbol{x}, y)$, and $D(\varepsilon)$.

Finally, (12) is scaled back to approximate the original normalizing constant $E(\boldsymbol{m})$.

B. Asymptotics

Without loss of generality, we consider (6) with $k = n$ and set $H(\boldsymbol{m}) = E(\boldsymbol{m})/\prod_{j=1}^{h} m_j!$ and $H_k(\boldsymbol{m}) = E_k(\boldsymbol{m})/\prod_{j=1}^{h} m_j!$, so that (6) may be rewritten as

$$H(\boldsymbol{m}) = H_n(\boldsymbol{m}) + \sum_j \gamma_{nj} H_n(\boldsymbol{m} - 1_j) \tag{13}$$

Let us now scale the capacity vector $\boldsymbol{m}$ and the number of items n by a parameter S and rewrite (13) as

$$h(\boldsymbol{x}, y) = h(\boldsymbol{x}, y-\varepsilon) + \sum_{j=1}^{h} g_j(y) h(\boldsymbol{x} - \varepsilon 1_j, y - \varepsilon) \tag{14}$$

where we define $\varepsilon = 1/S$, $y = n\varepsilon$, $x_j = m_j\varepsilon$, $\boldsymbol{x} = \boldsymbol{m}\varepsilon$, $x = \sum_j x_j$, and $h(\boldsymbol{x}, y)$ is a scaled $H(\boldsymbol{m})$ in which $\boldsymbol{x}$ correspond to list capacities and y to the number of item. Here $g_j(y)$ is assumed to be a smooth function defined such that $g_j(k\varepsilon) = \gamma_{kj}$ and $g_j(0\varepsilon) = g_j(n\varepsilon)$, $\forall j$. Note that by the above definitions, (14) reduces to (13) for $S = 1$.

The recurrence relation (14) can be solved asymptotically for $S \to \infty$, i.e., $\varepsilon \to 0$, using the singular perturbation method.

The formal derivation to obtain $\phi(\boldsymbol{x}, y)$, $B(\boldsymbol{x}, y)$, and $D(\varepsilon)$ from (14) is given in Appendix C, and leads to

$$\phi(\boldsymbol{x}, y) = \int_0^y \log\left(1 + \sum_{j=1}^{h} g_j(v)\xi_j\right) dv - \sum_{j=1}^{h} x_j \log \xi_j \tag{15}$$

where the terms ξ_j, $j = 1,\dots,h$ satisfy

$$x_j = \int_0^y \frac{g_j(v)\xi_j}{1+\sum_{l=1}^{h} g_l(v)\xi_l} dv \tag{16}$$

We show later in Section VI-A that the terms ξ_l may be seen as the asymptotic limits of the values $\xi_l(\boldsymbol{m})$ defined in (10). The remaining coefficients are $D(\varepsilon) = \varepsilon^{h/2}$ and

$$B(\boldsymbol{x}, y) = \frac{(2\pi)^{-h/2}}{\sqrt{\det \boldsymbol{J}}\sqrt{\xi_1 \cdots \xi_h}} \tag{17}$$

where the matrix $\boldsymbol{J} = [J_{jl}]$, $j, l = 1,\dots,h$, is given by

$$J_{jl} = \int_0^y \frac{\delta_{jl} g_j(v)}{1+\sum_r g_r(v)\xi_r} dv - \int_0^y \frac{\xi_j g_j(v) g_l(v)}{(1+\sum_r g_r(v)\xi_r)^2} dv \tag{18}$$

with $\delta_{jl} = 1$ if $j = l$, and $\delta_{jl} = 0$ otherwise.

C. Approximating a finite system

Having determined the asymptotics of $h(\boldsymbol{x}, y)$, we can now use the Euler-Maclaurin formula [36] to rewrite (12) in terms of the variables $\boldsymbol{m}$ and n. For a function $f(x)$ defined in $[0, n]$, the Euler-Maclaurin formula may be written as

$$\sum_{k=1}^{n} f(k) = \int_0^n f(x)dx + \frac{f(n) - f(0)}{2} + O(n^{-1})$$

Scaling variables in (16) so that the integral ranges in $[0, n]$, the error of the Euler-Maclaurin formula is $O(\varepsilon)$, and thus the resulting finite formula is asymptotically exact. Since we have chosen $g_j(0\varepsilon) \equiv g_j(n\varepsilon)$, this is given by

$$m_j = \sum_{k=1}^{n} \frac{\gamma_{kj}\xi_j}{1+\sum_{l=1}^{h}\gamma_{kl}\xi_l} \tag{19}$$

for $j = 1, \ldots, h$. Under uniform costs, [15] obtains an expression similar to (19) as the steady-state limit of mean field ordinary differential equations. Our derivation shows that a similar result holds under non-uniform costs.

Using Euler-Maclaurin we can also write

$$\phi(\boldsymbol{m}) = \sum_{k=1}^{n} \log\left(1+\sum_{j=1}^{h}\gamma_{kj}\xi_j\right) - \sum_{j=1}^{h} m_j \log \xi_j$$

and replace $\boldsymbol{J}$ with matrix $\boldsymbol{C} = [C_{jl}]$ where

$$C_{jl} = \sum_{k=1}^{n} \frac{\delta_{jl}\gamma_{kj}}{1+\sum_{r=1}^{h}\gamma_{kr}\xi_r} - \sum_{k=1}^{n} \frac{\xi_j\gamma_{kj}\gamma_{kl}}{(1+\sum_{r=1}^{h}\gamma_{kr}\xi_r)^2}$$

The above expressions provide by (12) the asymptotic expansion of $H(\boldsymbol{m})$, which implies the following expansion of the original normalizing constant

$$E(\boldsymbol{m}) \sim \frac{1}{(2\pi)^{h/2}} \frac{\prod_{k=1}^{n}\left(1+\sum_{j=1}^{h}\gamma_{kj}\xi_j\right)}{\prod_{j=1}^{h}\xi_j^{m_j+1/2}} \frac{\prod_{j=1}^{h} m_j!}{\sqrt{\det \boldsymbol{C}}} \tag{20}$$

where the ξ_j terms are obtained from (19).

D. Performance measures

For large $\boldsymbol{m}$, we can expand $E_k(\boldsymbol{m}-1_l)$ in a neighborhood of $\boldsymbol{m}$ to obtain by (20) the leading terms

$$\pi_{kl}(\boldsymbol{m}) \sim \begin{cases} \dfrac{1}{1+\sum_{j=1}^{h}\gamma_{kj}\xi_j} & \text{if } l=0 \\[2ex] \dfrac{\gamma_{kl}\xi_l}{1+\sum_{j=1}^{h}\gamma_{kj}\xi_j} & \text{if } l \geq 1 \end{cases} \tag{21}$$

for all $k = 1, \ldots, n$ and $l = 0, \ldots, j$. Additional terms in the expansions may also be used to generate higher-order approximations, which involve the derivatives of $\sqrt{\det \boldsymbol{C}}$. However, these are seldom needed since the cost of computing these terms is comparable to computing performance measures by direct application of (20) to the definitions given in Section III-C. We shall indicate the latter approach as the *singular perturbation approximation* (SPA), to distinguish it from the approach that numerically solves (19) for the ξ_l terms and then calculates performance measures by (21). The latter is further developed in the next section and referred to as the *fixed-point iteration* (FPI).

1) Example.: Table II illustrates the singular perturbation method. We consider a small-scale model where $E(\boldsymbol{m})$ can be computed numerically using (6). The model has $u = 2$ users, $\lambda_{1k}(l) = k^{-0.6}$, $\lambda_{2k}(l) = k^{-1.4}$, $\forall l$. We consider a linear structure for the cache and set access costs to $c_{vk}(l, l+1) = 1$, $l = 0, \ldots, h-1$. We fix $n = 2S$ and $m = S$ and use scalings $S = 2, 4, 8, 10$. As shown in the table, (20) converges to the exact value $E(\boldsymbol{m})$ as S grows.

TABLE II
EXAMPLE: NORMALIZING CONSTANT ASYMPTOTICS

n	m_1	m_2	$E(\boldsymbol{m})$	SPA - eq. (20)	relative error
4	2	0	$1.2969 \cdot 10^1$	$1.3691 \cdot 10^1$	0.056
8	4	0	$3.5950 \cdot 10^2$	$3.6940 \cdot 10^2$	0.028
16	8	0	$6.7136 \cdot 10^5$	$6.8063 \cdot 10^5$	0.014
20	10	0	$3.8500 \cdot 10^7$	$3.8926 \cdot 10^7$	0.011
4	1	1	$1.6173 \cdot 10^1$	$1.8919 \cdot 10^1$	0.170
8	2	2	$2.5697 \cdot 10^2$	$2.7810 \cdot 10^2$	0.082
16	4	4	$6.2439 \cdot 10^4$	$6.4990 \cdot 10^4$	0.041
20	5	5	$9.7236 \cdot 10^5$	$1.0042 \cdot 10^6$	0.033

VI. FURTHER APPROXIMATIONS

The goal of this section is to introduce an efficient fixed-point iteration to solve (19). We also develop simple bounds to quickly assess performance measures in settings where speed can be more important than high-accuracy, as in the numerical solution of NP-hard optimization programs.

A. Fixed-point iteration (FPI)

We now introduce a fixed-point method to obtain the ξ_l variables. Let us first consider a regular perturbation expansion [35] of (11), which for large m and n, with $m/n \sim O(1)$, takes the form $\pi_{il}(\boldsymbol{m}) \sim \hat{\pi}_{il}^{(0)} + \varepsilon\hat{\pi}_{il}^{(1)} + \ldots$, for small $\varepsilon > 0$. Plugging the last expression in (11), we get at the lowest order

$$\hat{\pi}_{il}^{(0)} = \frac{m_l\gamma_{il}(1-\sum_{j=1}^{h}\hat{\pi}_{ij}^{(0)})}{\sum_{k=1}^{n}\gamma_{kl}(1-\sum_{j=1}^{h}\hat{\pi}_{kj}^{(0)})} \tag{22}$$

for $i = 1, \ldots, n$, $l = 1, \ldots, h$. This is a non-linear system of equations with nh equations and nh unknowns. It is possible to verify that the system is solved by setting

$$\hat{\pi}_{il}^{(0)} = \frac{\gamma_{il}\xi_l^{(0)}}{1+\sum_{j=1}^{h}\gamma_{ij}\xi_j^{(0)}} \tag{23}$$

where $\xi_l^{(0)} = m_l(\sum_{k=1}^{n}\gamma_{kl}(1-\sum_{l=1}^{h}\hat{\pi}_{il}^{(0)}))^{-1}$. Applying the regular perturbation expansion to the terms in (2) and plugging (23) in the resulting expression, we see that as $\varepsilon \to 0$ the $\xi_l^{(0)}$ terms become solutions of (19). Since $\xi_l^{(0)}$ is the value of ξ_l corresponding to the leading terms of the expansion of $\pi_{il}(\boldsymbol{m})$, this provides support to our earlier statement that the variables ξ_l appearing in the expansion of $E(\boldsymbol{m})$ are the limits of $\xi_l(\boldsymbol{m})$ for $\varepsilon \to 0$. A similar conclusion follows by calculating the leading term of (7) as in Section V-D.

The explicit expression of $\xi_l^{(0)}$ enables us to solve (19) by introducing a fixed point iteration over the marginal probabilities. This is given by

$$\xi_l^{(t+1)} = \frac{m_l}{\sum_{k=1}^{n}\gamma_{kl}(1-\sum_{l=1}^{h}\pi_{il}^{(t)})} \tag{24a}$$

$$\pi_{il}^{(t+1)} = \frac{\gamma_{il}\xi_l^{(t+1)}}{1+\sum_{j=1}^{h}\gamma_{ij}\xi_j^{(t+1)}} \tag{24b}$$

for all $i = 1, \ldots, n$, $l = 1, \ldots, h$ and iterations $t \geq 0$. For uniform costs, the above expression correctly reduces to the ones obtained via mean field approximation in [15].

As mentioned before, the FPI method solves (19), which can be done by (24), and then plugs the obtained ξ_l values

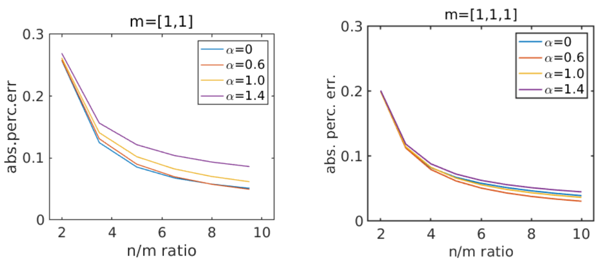


Fig. 2. Approximation error in $[0,1]$ of (24) for the miss rate $M(\boldsymbol{m})$ in two-stream caches with $h=2$ and $h=3$ lists, with convergence tolerance $\tau=10^{-6}$. The cache has two users with request rates $\lambda_{1k}(l)=p_k$, $p_k=k^{-\alpha}$, $k=1,\ldots,n/3$, $\lambda_{2k}(kl)=q_k$, $q_k=k^{-\alpha}$, $k=(n/3+1),\ldots,n$, and $\lambda_{1k(l)}=\lambda_{2k(l)}=0$ otherwise, $\forall l$. Costs are set to unity. The cache structure is linear.

into (21) to obtain performance measures. The iteration can be initialized with a guess of the marginal probabilities, e.g., $\pi_{il}^{(0)}=1/(h+1)$. We stop it when the maximum relative change of the miss ratios $\pi_{i0}^{(t)}=1-\sum_l \pi_{il}^{(t)}$ across iterations does not exceed a user-specified tolerance (e.g., $\tau=10^{-6}$).

In the over 2,000 models considered in the experimental section, none raised convergence issues. However, in contrived examples it is possible to show particular assignments of the access costs which result in the above iteration not to converge. In such situations, one may seek to use a more general nonlinear equation solver to find a set of values for ξ_l and π_l which satisfy the equations.

1) Example.: Figure 2 evaluates the error in computing the miss ratio via fixed-point iteration against the exact solution. Let $\widehat{M}(\boldsymbol{m})$ be the approximate value obtained for the miss rate. The error metric shown in the figure is the absolute percentage error

$$\epsilon_{mape}=\left|1-\frac{\widehat{M}(\boldsymbol{m})}{M(\boldsymbol{m})}\right| \tag{25}$$

We here consider two caches with $h=2$ and $h=3$ lists and $u=2$ streams. The streams request two independent sets of items, both according to a Zipf-like popularity distribution with parameter α.

As the ratio n/m grows, we see that the cache approaches the asymptotic behavior and the error visibly decreases. It is clear from the example that considering models with a larger number of lists h also reduces error, for fixed ratio n/m, and makes the results less sensitive to the exponent α. The maximum number of iterations of (24a)-(24b) is observed to be $t_{max}=23$ for $\alpha=0$, and $t_{max}=36$ for $\alpha=1.4$. This means in practice that the iteration converges within the tolerance in just a few milliseconds. We also observe the number of iterations to slightly decrease as n/m grows. These results are representative also of other parameterizations. Experiments with larger models are reported in Section VII.

B. Bound analysis

Here we develop some simple analytical bounds on $\xi_l(\boldsymbol{m})$ and π_{i0}. Bounds on $\xi_l(\boldsymbol{m})$ allow us to bound by (8) the miss rate $M(\boldsymbol{m})$, whereas bounds on the probabilities π_{i0} can be used to bound the miss rates using (5).

1) Bounds on $\xi_l(\boldsymbol{m})$: It is possible to verify that

$$\frac{m_l}{\gamma_l^+(n-m+1)}\le \xi_l(\boldsymbol{m})\le \frac{m_l}{\gamma_l^-(n-m+1)} \tag{26}$$

where $\gamma_l^+=\max_k \gamma_{kl}$ and $\gamma_l^-=\min_k \gamma_{kl}$. These follow from (10) by bounding γ_{kl} with γ_l^+ or γ_l^- and noting that $\sum_k(1-\pi_k(\boldsymbol{m}-1_l))=n-m+1$. For an important class of models, we then state a tighter lower bound. Lower bounds on $\xi_l(\boldsymbol{m})$ are the most important in practice, as they lead to pessimistic estimates of the miss rates. The bound assumptions are satisfied by a rather broad class of models including, but not limited to, models with uniform costs.

Theorem 3. *Consider a cache where for each pair of items i and $k>i$, we have $\gamma_{il}\ge\gamma_{kl}$, $\forall l$. Then*

$$\frac{m_l}{\bar{\gamma}_l(n-m+1)}\le \xi_l(\boldsymbol{m}) \tag{27}$$

where $\bar{\gamma}_l=\sum_j \gamma_{jl}/n$ is the average cost factor for list l.

The proof is given in Appendix D and shows that the argument generalizes for any list where $\gamma_{kl}\ge\gamma_{il}$ implies $\pi_k(\boldsymbol{m}-1_l)\ge\pi_i(\boldsymbol{m}-1_l)$. If the property holds for all lists, we say that the model is *monotone*.

2) Bounds on $\pi_{k0}(\boldsymbol{m})$: We now develop upper bounds on the miss rate $\pi_{k0}(\boldsymbol{m})$. Let $\xi_j^{-k}(\boldsymbol{m})$ be defined as $\xi_j(\boldsymbol{m})$, but for a model without item k. We begin by establishing two general monotonicity properties.

Lemma 1. $\xi_j(\boldsymbol{m}-1_l)\le\xi_j(\boldsymbol{m})\le\xi_j^{-k}(\boldsymbol{m})$, $\forall j,l,k$.

Lemma 2. $\pi_{k0}(\boldsymbol{m})\le\pi_{k0}(\boldsymbol{m}-1_l)$, $\forall k,l$.

The proofs are given in Appendix E and Appendix F. We are now ready to give an upper bound on the miss rates.

Theorem 4. *The item miss ratios satisfy the bound $\pi_{k0}(\boldsymbol{m})\le\left(1+\sum_{j=1}^h\gamma_{kj}\xi_j(\boldsymbol{m})\right)^{-1}$ for all items $k=1,\ldots,n$ and lists $l=1,\ldots,h$.*

The proof is given in Appendix G. Note that bound is asymptotically exact for fixed h, as it converges to (21) when n and m increase in a fixed ratio. A closed-form expression for this upper bound can be readily obtained by replacing $\xi_l(\boldsymbol{m})$ with the lower bound in either (26) or (27). Although (27) is not guaranteed to be a bound in non-monotone models, combining it with Theorem 4 still provides a useful non-iterative heuristic approximation of π_{k0}, as illustrated in the next example.

3) Example.: Figure 3 illustrates Theorem 4 for a monotone and a non-monotone cache. Estimates are computed by replacing $\xi_j(\boldsymbol{m})$ in the bound of Theorem 4 with (27). In monotone caches, which include caches with uniform cost as a special case, rank is inverse proportional to item miss rates [15]. However, this clearly does not hold in non-monotone caches, as shown in the figure. As visible from comparison against the exact value, the heuristic provides rather accurate estimates of the item miss ratio.

VII. Numerical results

A. Simulation-based validation

Since exact numerical solution is applicable only to small models, we have performed a simulation study of RR-c($\boldsymbol{m}$)

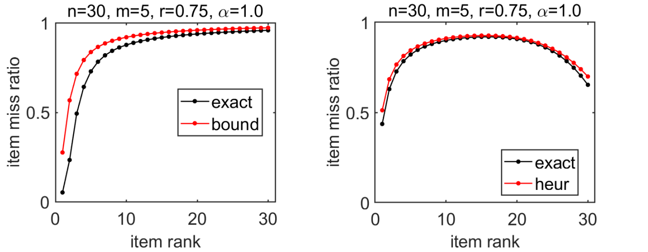


Fig. 3. Single-stream cache with capacity $\boldsymbol{m} = (3, 2)$ and request rates $\lambda_{vk}(l) = p_k\lambda$, with $\lambda = 1$ and Zipf-like popularity $p_k = k^{-\alpha}$. Items are ranked according to p_k. The left figure shows a monotone cache with $c_{vk}(l, l+1) = (1-r)r^k$, the right cache is non-monotone with costs c_{vkl}^{-1}.

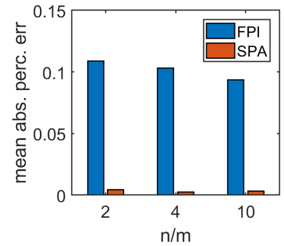
TABLE III
SIMULATION PARAMETERS FOR RR-C($\boldsymbol{m}$), FIFO-C($\boldsymbol{m}$)

n	Number of items	10, 1000
n/m	Items-to-capacity ratio	2, 4, 10
h	Number of lists	1, 2, 5
u	Number of streams	1, 4
α	Zipf parameter	0.6, 1.0, 1.4
c	Normalized access cost	0.1, 0.5, 1.0

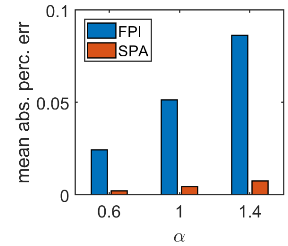
and FIFO-c($\boldsymbol{m}$) to assess the accuracy of the proposed approximations. The set of parameters used in the study is given in Table VII. Whenever $n = m$, we set $n = m + 2$ to avoid degeneracies. For comparison with prior work, we focus on this round of experiments on linear caches as in Figure 1(b) and where $c_{vi}(0, e) = 1$, where $e = 1$, and $c_{vi}(1, l+1) = \ldots = c_{vi}(l, l+1) = c$, with $c = 0.1, 0.5, 1.0$. It is useful to note that, because of the way γ_{ij} is defined in (1), models with identical access costs c for all lists $l \geq 1$ must have identical equilibrium state probabilities. Our rationale for including sensitivity with respect to c is that the fixed-point equation (19) is not invariant to c, so we show that this however does not affect the results of FPI and SPA.

Access costs are set identically for each list and equal to c. Each stream sends Poisson requests at rate $\lambda = 1$, scaled by a Zipf-like distribution with parameter α, with the object ranks assigned uniformly at random within each stream. Each simulation collects 10^9 state samples from the cache. From the results, we observe as expected that RR-c($\boldsymbol{m}$) and FIFO-c($\boldsymbol{m}$) have statistically indistinguishable performance, thus we discuss only RR-c($\boldsymbol{m}$) and compare the FPI and SPA methods of Section V-D, based on the mean absolute percentage error for the item miss ratios $\pi_{i0}(\boldsymbol{m})$. Note that the latter metric gives equal weight to each item, therefore assessing the ability of the methods to accurately approximate miss ratios for both popular and rarely accessed items.

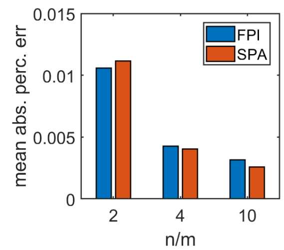
In small models with $n = 10$ items, SPA incurs a mean absolute percentage error lower than FPI, 0.4% instead of 10.2%. For such instances, we have observed the maximum error of FPI to reach 35.1%, whereas for SPA it is just 0.6%. We have observed the largest errors of FPI to be in the estimation of miss probabilities for high popularity items. Conversely, with $n = 1000$ items the cache dynamics reaches the asymptotic regime and as expected both methods converge with a mean absolute percentage error of 0.6% for both methods, and a maximum just below 0.7% error for both methods.



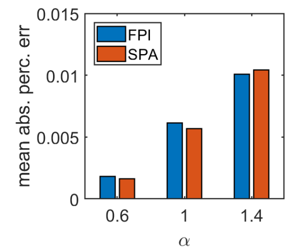
(a) $n = 10$



(b) $n = 10$



(c) $n = 1000$



(d) $n = 1000$

Fig. 4. Sensitivity of FPI and SPA to changes in the item-to-capacity ratio n/m and in the parameter α for caches with $n = 10$ items and $n = 1000$ items.

We have also investigated the sensitivity of the results to the simulation parameters. We have noticed the results to be nearly insensitive to changes in the access cost c and the number of streams u. Figure 4 illustrates the dependence of the error on the ratio n/m and α. Let $\widehat{\pi}_{0k}(\boldsymbol{m})$ be the estimated item miss ratio for item k. Then the error metric shown in the figure here and in similar comparisons throughout this section is

$$\epsilon_{mape}^{\pi} = \frac{1}{n}\sum_{k=1}^{n}\left|1 - \frac{\widehat{\pi}_{0k}(\boldsymbol{m})}{\pi_{0k}(\boldsymbol{m})}\right| \quad (28)$$

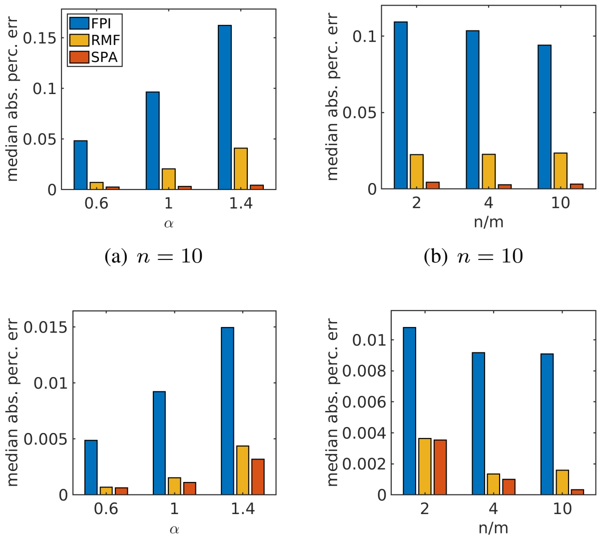
Note that since we take the average after calculating the error for each item, this metric weights uniformly relative errors across the whole item set.

Both approximations tend to be more accurate in caches with a higher n/m ratio, thus we expect our methods to perform accurately in models where a small fraction of the items can be cached. Some dependence is observed with respect to increasing value of α, however the implied error becomes rather small in all cases if $n = 1000$. We have observed a similar behavior also under changes of the number of lists h, where accuracy is generally better for smaller values of h, but errors become negligible if $n = 1000$.

Overall, the results indicate that FPI suffices in practice on realistic models, but SPA provides more robust estimates for small models far from the asymptotic regime.

B. Comparison with a refined mean-field approximation

We now further validate the accuracy of SPA against the refined mean-field approximation (RMF) developed in [37]. The latter is applied to linear caches similar to the ones studied in the previous section. The refined mean field was originally developed in [37] for systems of N homogeneous objects. It



(a) $n = 10$ (b) $n = 10$

(c) $n = 100$ (d) $n = 100$

Fig. 5. Comparison between FPI, SPA and refined mean-field approximation (RMF) for linear caches with $n = 10$ items and $n = 100$ items.

differs from standard mean-field approximations by introducing a correction term that significantly increases the accuracy of the analysis for small N. In this paper, we construct the refined mean field equation as if there were N identical copies of the original n objects. We then apply the correction term after setting $N = 1$. Appendix M gives details about the methodology followed to derive the RMF equations for this type of caches.

Approximation results for $n = 10$ and $n = 100$ are given in Figure 5. We compare FPI, which in fact gives for this system the same solution as the classical mean field approximation, the refined mean field (RMF) and FPI. As already visible at $n = 100$ the RMF and FPI converge to results of similar accuracy, below 0.5% error. However, for small caches with $n = 10$ objects, SPA features errors of smaller magnitude than the other techniques.

Since caches have typically a large number of objects, we conclude that both RMF and SPA provide an accurate way to analyze list-based caches. Main differences are that SPA has a compact expression and does not require to use ordinary differential equations (ODEs) and thus does not incur issues such as ODE stiffness or implementation-specific overheads associated to ODE setup and solution, particularly for large systems. Conversely, RMF has the potential to be extended to transient regime, whereas SPA is inherently limited to equilibrium solutions.

C. Youtube trace

Lastly, we use a trace-driven simulation to illustrate the applicability of the proposed methods to real workloads. We have simulated the 2008 Youtube trace collected in [38] and also used to validate the list-based cache models in [15], [26]. The trace consists of $611,968$ requests, spanning $n = 303,332$ items and originating from $u = 16,337$ client IPs. Each client IP is mapped to a single stream v.

We run trace-driven simulations, using caches with capacity $m = 5000$, $m_1 = 2900$, and $m_2 = \cdots = m_h = (m - m_1)/(h-1)$ and varying the number of lists $h = 2, 3, 5$, the replacement policy (FIFO-c($\boldsymbol{m}$), RR-c($\boldsymbol{m}$)), and the normalized access costs to the next list ($c = 0.1, 0.5, 1.0$).

For each simulation, we compute the item-miss rates M_k, $k = 1 \ldots, n$, and compare them with the estimates returned by FPI. In the stochastic model, each stream v is modeled as a Poisson process with rate $\lambda_v(k) = n_{vk}/T$, where n_{vk} is the number of times item k is requested by stream v and T is the timespan of the trace.

Despite the large number of items, FPI produces an estimate of all item miss rates in just 8.2 seconds on a laptop (Intel Core i5). Even though the Youtube trace is temporally-correlated, thus departing from IRM and Poisson assumptions, and about half of the items are accessed just a single time each, the mean absolute percentage error of FPI on the M_k values is only 1.74%, with a maximum error of just 4.68%. Both values are compatible with the uncertainties of the simulation, in which the trace is simulated only once. Overall, the experiments indicate that our model can reliably reproduce simulation results in the presence of access costs and multiple request streams.

VIII. Synchronous streams

In this section, we modify the reference model to handle streams that send synchronous requests, i.e., each stream waits for a reply from the cache before sending the next request. The model may therefore be seen as closed, with a set of circulating requests to the cache that periodically require some think time in between new arrivals. Models of this kind are useful to evaluate the impact of cache replacement policies under pooled cache access.

A. Reference model

We now assume that each stream can have up to s_v outstanding synchronous requests. The stream requires a think time with mean σ_v between completion of a request and issue of the next one; we assume this think time to be independent of the cache state. If the request generates a cache miss, then the miss latency is on average θ_{v0}, otherwise the hit latency is θ_{v1}. We assume think times and hit/miss latencies to follow an exponential distribution. We also assume that the cache contents are updated upon request arrival and that concurrent requests at the cache are served in parallel without contention.

B. Iterative method

We now leverage the analytical results developed in the earlier sections to develop an iterative approximation for this model. Let $\Lambda_v = \sum_k \Lambda_{vk}$ be the total arrival rate from stream v, which is an unknown to be determined. By Little's law, the following relationship holds for each stream $v = 1, \ldots, u$

$$s_v = \sum_{k=1}^{n} \Big(\Lambda_{vk}\sigma_v + M_{vk}\theta_{v0} + H_{vk}\theta_{v1} \Big)$$

where $H_{vk} = \Lambda_{vk} - M_{vk}$ is the hit rate for item k and stream v. Each term corresponds to the average number of requests either receiving a delay σ_v, θ_{v0}, or θ_{v1}.

Let $H_v = \sum_{k=1}^{n} H_{vk}$, we may rewrite the last equation as

$$\Lambda_v = \frac{s_v}{\sigma_v + (M_v/\Lambda_v)\theta_{v0} + (H_v/\Lambda_v)\theta_{v1}} \tag{29}$$

Starting from an initial guess $\Lambda_v^{(0)}$ for the rates, a fixed-point iteration may be used to solve the previous equation, after rearranging terms as

$$\Lambda_v^{(t+1)} = \frac{s_v \Lambda_v^{(t)}}{\Lambda_v^{(t)}\sigma_v + M_v^{(t)}\theta_{v0} + H_v^{(t)}\theta_{v1}} \tag{30}$$

for $t \geq 0$. Using a decomposition approximation, and assuming that a guess $\Lambda_v^{(0)}$ is available for the stream arrival rates, we can readily estimate $M_v^{(0)}$ and $H_v^{(0)}$ by obtaining the corresponding quantities from a model of an isolated cache fed by Poisson streams with rates $\Lambda_v^{(0)}$ and identical item reference probabilities as in the original model. Solving this isolated model using FPI or SPA, we can then apply (30) to produce a new estimate $\Lambda_v^{(1)}$ for the stream arrival rates in the initial model. By iterating this scheme, we can obtain successive approximations for the miss and hit rates $M_v^{(t)}$ and $H_v^{(t)}$ at iteration t and for the arrival rates $\Lambda_v^{(t)}$. We can then obtain the average cache throughput for requests from stream v by observing that this matches at equilibrium the arrival rate Λ_v.

We also remark that the proposed analysis method can be readily extended to caching processing both synchronous and asynchronous streams. Let u' be the number of asynchronous streams, then upon solving the cache in isolation it is sufficient to include u' additional Poisson streams Λ'_v, $v = 1, \ldots, u'$, with the same arrival rates and item reference models as the asynchronous streams.

C. Results

We have verified the accuracy of the proposed iterative approximation using a set of 810 experiments with the parameters given in Appendix N1 in the supplemental material, together with additional validation results. In the models, α_v describes the Zipf parameter for stream v. Each simulation collects 10^7 samples. Iterations are stopped when the maximum absolute relative difference between the predicted Λ_v values in successive iterations falls below 10^{-3}. Prediction errors for FPI and SPA obtained for FIFO-$c(\boldsymbol{m})$ and RR-$c(\boldsymbol{m})$ against simulation when setting the initial rate guesses to an arbitrary value ($\Lambda_v^{(0)} = 1.0$, $\forall v$) are shown in Figure 6.

The results indicate that the decomposition approximation is accurate with both FPI and SPA. The overall mean percentage error on the cache throughputs Λ_v is 1.64% for FPI and 0.25% for SPA. The approximation error is generally consistent with the trends observed in Figure 4 for asynchronous requests.

Generally the larger the cache or the number of lists, the more accurate the approximations become. The 3.30% error for FPI in the case $n/m = 10$ is due to a relatively higher error on degenerate models with $n = 10$ and $m = 1$, in which FPI incurs a 7.21% mean absolute percentage error when $\alpha = 1.4$, whereas this is just 1.65% for SPA.

We also see that think times do not significantly affect the precision of the method. Further inspection reveals that, as for asynchronous requests, errors are moderately increasing

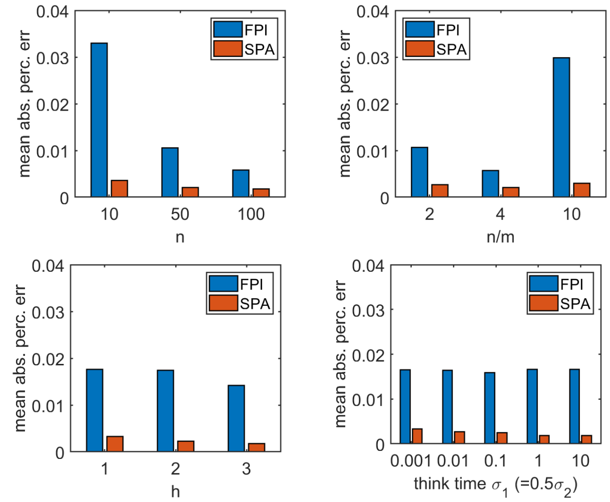


Fig. 6. FPI and SPA item miss ratios for models with synchronous requests.

under increasing α_1. That is, FPI goes from 1.26% error for $\alpha_1 = 0.6$ to 2.00% error for $\alpha_1 = 1.4$, while in the same range the mean error for SPA grows from 0.22% to 0.29%.

Furthermore, we observe nearly identical accuracy for FIFO-$c(\boldsymbol{m})$ and RR-$c(\boldsymbol{m})$, within a 10^{-4} tolerance. Conversely, when simulating a cache policy such as LRU-type policy[1] as expected the error of the proposed technique grows significantly, as it is not intended to cope with this replacement policy. For example, the maximum error for SPA with LRU in models with $h = 1$ and $\alpha_1 = 1.4$ is 17.56%. This further confirms that the model is able to capture the distinctive features of RR-$c(\boldsymbol{m})$ and FIFO-$c(\boldsymbol{m})$.

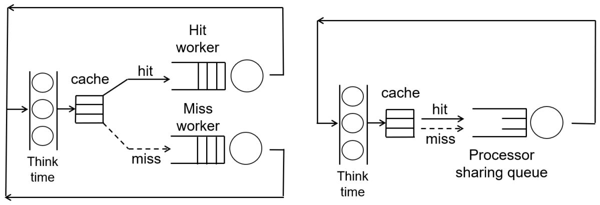
Additional sensitivity experiments with respect to the number of outstanding requests and to the access costs and tree structure are given in Appendix N.

1) Summary of validation.: Overall, the above results indicate that models with synchronous streams can be accurately analyzed using the FPI and SPA approximations, respectively with mean errors of 1.64% and 0.25%. Since the proposed decomposition only relies on average information, and implicitly assumes that the arrivals at the isolated cache form a Poisson process, it seems possible to conclude that the rate estimates for the different streams are sufficient to characterize the model performance behavior under synchronous requests.

IX. Queueing analysis

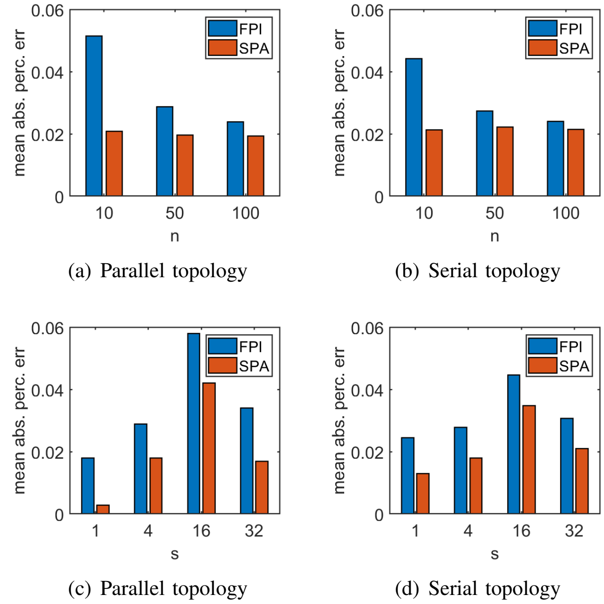
We now demonstrate the applicability of FPI and SPA to models featuring queueing. An example of the class of models studied in this section is given in Figure 7(a). Requests cycle between an infinite server node, used to represent think times, and workers (i.e., queueing stations) that handle processing of the cache requests. The distinctive feature of the model is that requests visiting cache nodes change class according

[1]We may define the LRU-c($\boldsymbol{m}$) policy as derived from FIFO-c($\boldsymbol{m}$), with the difference that the victim item k is moved to the front of list l, shifting items in l back by one position up to filling the position formerly held by i. An hit on item i in position j in a leaf list l' moves i to the position 1 in l', shifting back by one position all items occupying positions $1, \ldots, j-1$.



(a) Parallel topology (b) Serial topology

Fig. 7. Examples of stochastic networks including a cache and queueing stations. Routing and class-switching are state-dependent, since they depend on cache hits and cache misses.



(a) Parallel topology (b) Serial topology

(c) Parallel topology (d) Serial topology

Fig. 8. FPI and SPA item miss ratios on the models with queueing shown in Figure 7.

to whether they find or not the requested items in the cache. Since routing is class-dependent and visits to the cache are instantaneous, this behavior may also be seen as a form of state-dependent routing. In this section, we formalize this class of models and propose an iterative approximation method.

A. Model

To represent models such as the one in Figure 7(a), we introduce a class of extended queueing networks featuring u synchronous streams, u' asynchronous streams, and p processing nodes, either queueing stations or caches. Stations can use any scheduling policy supported by approximate mean-value analysis (AMVA) [39], e.g., infinite server, processor sharing, first-come first-served. As in the previous section, our goal is to determine the mean throughput Λ_v for synchronous streams, assuming that the number of outstanding requests is at most s_v for stream v.

Each stream v is associated to a chain K_v. A chain consists of a set of class labels, such that a request from v can travel through the network only as a request in some class $c \in K_v$. Chains are assumed to be non-overlapping. Each chain is associated to a *reference node*, chosen such that passage of the request through it represents a request completion. Furthermore, a request from stream v arriving in class k at cache node i asks for a single item and then leaves in class hit_{ik} after a cache hit and in class $miss_{ik}$ after a cache miss. The definition of classes $hit_{ik} \in K_v$ and $miss_{ik} \in K_v$ for every tuple (v, i, k) is a model input parameter. We further assume that a request in class k takes an exponentially distributed time with mean latency θ_{ik} to complete at station i, inclusive of the average visit count. We require in particular that $\theta_{ik} = 0$ if i is a cache, so that caches are traversed instantaneously.

B. Iterative method

To analyze the proposed class of models, we define a modification of the Bard-Schweitzer AMVA that relies on SPA or FPI to analyze cache nodes. In this approximation, we consider a modified queueing network where the cache nodes are replaced by infinite servers, with infinite service rate in all classes, and in which the routing probability $\gamma'_{i,k,j,s}$ for a job that arrives in class k at cache i to later leave towards station j in class s is set to

$$\gamma'_{i,k,j,s} = \frac{M_{ik}}{\Lambda_{ik}} \gamma_{i,miss_{ik},j,s} + \frac{H_{ik}}{\Lambda_{ik}} \gamma_{i,hit_{ik},j,s} \tag{31}$$

where $\gamma_{i,k,j,s}$ are the state-independent probabilities in the original model with the caches, M_{ik} is the miss rate for items requested by class k at cache node i, $H_{ik} = \Lambda_{ik} - M_{ik}$ and Λ_{ik} are the corresponding hit rate and request arrival rate. As in Section VIII-B, the miss rates M_{ik} can be determined by FPI or SPA by considering the cache nodes in isolation with Poisson arrival rates Λ_{ik}.

Using (31), a state-independent probability routing matrix is determined for the closed network, from which it is possible to obtain average visit ratios and thus determine the mean latencies θ_{ik} at every station. Since we assume infinite rates at cache nodes, their mean latencies will be set to zero, however their behavior is reflected in the routing matrix.

With the obtained latencies, we can apply AMVA to compute the throughputs Λ_v for the requests issued by each synchronous stream $v = 1, \ldots, u$, assuming a total job population s_v in class v that starts at the reference station. We here consider in particular AMVA with standard extensions for chains analysis [39].

The above scheme can be applied iteratively, so that the throughputs $\Lambda_{ik}^{(t)}$ determined by AMVA at iteration t provide the input arrival rates to each cache node, which is then solved in isolation by FPI or SPA to produce updated miss rates $M_{ik}^{(t)}$ and hit rates $H_{ik}^{(t)}$ values at the same iteration. The latter values are then used to update the routing probabilities in (31) at the following iteration. Upon terminating, this method returns an approximate value for the arrival rates and throughputs Λ_{ik} of each stream at the reference station.

C. Numerical examples

To illustrate the accuracy of the method, we consider the model in Figure 7(a)-(b), with the parameters reported in Appendix O1 of the supplemental material, together with additional validation results. Each parameterization is evaluated with 216 simulations, each taking 10^7 samples, for a total of 648 simulations.

1) Parallel topology.: We consider first the model in Figure 7(a). The two streams can issue at most $s_1 = s_2 = s$ synchronous requests each. We run the proposed iterative method using 10^{-3} tolerance on the maximum absolute percentage error on the arrival rates Λ_{iv} at all nodes. Simulation uses 10^7 samples. Numerical results are shown in figures 8(a) and 8(c). Errors are moderately larger than in earlier experiments, but we can also see in this case that as the cache size increases the approximation displays decreasing errors. In addition, we notice a dependence between error and maximum number of circulating requests, which demonstrates that both FPI and SPA are accurate especially in light and heavy load, with errors below 4% for FPI and 2% for SPA.

2) Serial topology.: We now switch our attention to the model in Figure 7(b). In this case, job hits or misses determine the mean service time at the processor sharing queueing. Results are obtained as in the last example and shown in figures 8(b) and 8(d). Trends are qualitatively similar, with errors marginally lower than the parallel topology case.

3) Summary.: The above examples demonstrate that fixed-point iterations based on FPI and SPA are effective in approximating the performance of queueing in the presence of caches. To the best of our knowledge, no similar approximations have been studied in earlier work, implying that the algorithms in Section IX as the first class of approximations for stochastic networks including both caches and queueing stations.

X. Conclusion

We have generalized models for list-based caches to include non-uniform access costs. After solving the Markov process for the RR and FIFO policies, we have developed exact and approximate formulas to efficiently analyze performance measures such as miss rates. We have then applied singular perturbation methods to determine the asymptotic behavior of the cache. Simulation results indicate that our approximations typically incur less than 1% error on large models. We have further proposed to use this result to approximate caches with synchronous calls or extended queueing networks with caching nodes, obtaining similar levels of accuracy.

References

[1] S. Podlipnig and L. Boszormenyi, "A survey of web cache replacement strategies," *ACM Computing Surveys*, vol. 35, 2003.

[2] M. M. Amble, P. Parag, S. Shakkottai, and L. Ying, "Content-aware caching and traffic management in content distribution networks," in *Proc. INFOCOM*, pp. 2858–2866, 2011.

[3] N. Carlsson and D. L. Eager, "Ephemeral content popularity at the edge and implications for on-demand caching," *IEEE TPDS*, vol. 28, no. 6, pp. 1621–1634, 2017.

[4] O. Saleh and M. Hefeeda, "Modeling and caching of peer-to-peer traffic," in *Proc. ICNP*, pp. 249–258, IEEE, 2006.

[5] R. Fagin, "Asymptotic miss ratios over independent references," *J. Comp. and Sys. Sci.*, vol. 14, pp. 222–250, Apr. 1977.

[6] R. Fagin and T. G. Price, "Efficient calculation of expected miss ratios in the independent reference model," *SIAM Journal on Computing*, vol. 7, 1978.

[7] E. Gelenbe, "A unified approach to the evaluation of a class of replacement algorithms," *IEEE Trans. Computers*, vol. 22, no. 6, pp. 611–618, 1973.

[8] O. I. Aven, E. G. Coffman, and Y. A. Kogan, *Stochastic analysis of computer storage*. Reidel, 1987.

[9] J. V. D. Berg and A. Gandolfi, "LRU is better than FIFO under the independent reference model," *J. App. Prob.*, vol. 29, no. 1, pp. 239–243, 1992.

[10] K. Psounis and B. Prabhakar, "Efficient randomized web-cache replacement schemes using samples from past eviction times," *IEEE/ACM Trans. Netw*, vol. 10, no. 4, pp. 441–455, 2002.

[11] N. Tsukada, R. Hirade, and N. Miyoshi, "Fluid limit analysis of FIFO and RR caching for independent reference models," *PEVA*, vol. 69, no. 9, pp. 403–412, 2012.

[12] J. Li, S. Shakkottai, J. C. S. Lui, and V. Subramanian, "Accurate learning or fast mixing? Dynamic adaptability of caching algorithms," *CoRR*, vol. abs/1701.02214, 2017.

[13] N. Carlsson and D. L. Eager, "Worst-case bounds and optimized cache on mth request cache insertion policies under elastic conditions," *PEVA*, 2018.

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

[15] N. Gast and B. V. Houdt, "Transient and steady-state regime of a family of list-based cache replacement algorithms," in *Proc. SIGMETRICS*, pp. 123–136, ACM, 2015.

[16] A. Aho, P. Denning, and J. Ullman, "Principles of optimal page replacement," *JACM*, vol. 18, 1971.

[17] W. F. King, "Analysis of demand paging algorithms," in *IFIP Congress (1)*, pp. 485–490, 1971.

[18] L. Shi, X. Wang, R. T. B. Ma, and Y. C. Tay, "Weighted fair caching: Occupancy-centric allocation for space-shared resources," *PEVA*, pp. 194–211, 2018.

[19] C. Knessl and C. Tier, "Asymptotic expansions for large closed queueing networks with multiple job classes," *IEEE Trans. Computers*, vol. 41, no. 4, pp. 480–488, 1992.

[20] D. S. Berger, P. Gland, S. Singla, and F. Ciucu, "Exact analysis of TTL cache networks," *Perform. Eval*, vol. 79, pp. 2–23, 2014.

[21] H. Che, Y. Tung, and Z. Wang, "Hierarchical web caching systems: modeling, design and experimental results," *IEEE JSAC*, vol. 20, no. 7, pp. 1305–1314, 2002.

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

[23] N. Gast and B. V. Houdt, "Asymptotically exact TTL-approximations of the cache replacement algorithms LRU(m) and h-LRU," in *Proc. ITC*, pp. 157–165, IEEE, 2016.

[24] A. Dan and D. F. Towsley, "An approximate analysis of the LRU and FIFO buffer replacement schemes," in *Proc. SIGMETRICS*, pp. 143–152, ACM, 1990.

[25] M. Gallo, B. Kauffmann, L. Muscariello, A. Simonian, and C. Tanguy, "Performance evaluation of the random replacement policy for networks of caches," *PEVA*, vol. 72, pp. 16–36, 2014.

[26] G. Ju, Y. Li, Y. Xu, J. Chen, and J. C. S. Lui, "Stochastic modeling of hybrid cache systems," in *Proc. MASCOTS*, pp. 69–78, IEEE, 2016.

[27] D. Starobinski and D. Tse, "Probabilistic methods in web caching," *PEVA*, vol. 46, pp. 125–137, Oct. 2001.

[28] O. Bahat and A. M. Makowski, "Optimal replacement policies for non-uniform cache objects with optional eviction," in *Proc. INFOCOM*, pp. 427–437, IEEE, 2003.

[29] G. Neglia, D. Carra, M. Feng, V. Janardhan, P. Michiardi, and D. Tsigkari, "Access-time aware cache algorithms," in *Proc. of ITC*, pp. 148–156, IEEE, 2016.

[30] G. Casale, "Analyzing replacement policies in list-based caches with non-uniform access costs," in *Proc. INFOCOM*, pp. 432–440, IEEE, 2018.

[31] F. P. Kelly, *Reversibility and Stochastic Networks*. New York: John Wiley & Sons, 1979.

[32] S. Lam, "Dynamic scaling and growth behavior of queueing network normalization constants," *JACM*, vol. 29, no. 2, pp. 492–513, 1982.

[33] M. Reiser and S. S. Lavenberg, "Mean-value analysis of closed multichain queueing networks," *JACM*, vol. 27, no. 2, pp. 312–322, 1980.

[34] J. Zahorjan, "The distribution of network states during residence times in product form queueing networks," *Perform. Eval*, vol. 4, no. 2, pp. 99–104, 1984.

[35] C. Knessl and C. Tier, "Applications of singular perturbation methods in queueing," in *Frontiers in Queueing: Models and Applications in Science and Engineering*, pp. 331–336, CRC Press, 1996.

[36] E. W. Weisstein, *CRC Concise Encyclopedia of Mathematics*. Chapman and Hall, 2003.

[37] N. Gast and B. V. Houdt, "A refined mean field approximation," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 1, no. 2, p. 33, 2017.
[38] M. Zink, K. Suh, Y. Gu, and J. Kurose, "Characteristics of YouTube network traffic at a campus network - Measurements, models, and implications," *Computer Networks*, vol. 53, no. 4, pp. 501–514, 2009.
[39] *et al.*. G. Bolch, *Queueing Networks and Markov Chains*. Wiley, 2006.
[40] R. Courant and D. Hilbert, *Methods of Mathematical Physics, Volume II*. Interscience, 1962.
[41] G. H. Hardy, J. E. Littlewood, and G. Pólya, *Inequalities*. CUP, 1952.
[42] N. Gast, "Expected values estimated via mean-field approximation are 1/n-accurate," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 1, no. 1, p. 17, 2017.
[43] N. Gast, "Construction of lyapunov functions via relative entropy with application to caching," *ACM SIGMETRICS PER*, vol. 44, no. 2, pp. 6–8, 2016.
[44] N. Gast, "Size expansions of mean field approximation: Transient and steady-state analysis," *Performance Evaluation*, 2018.

Appendix

A. Proof of product-form solution

1) RR-c($\boldsymbol{m}$) policy.: Under RR-c($\boldsymbol{m}$), consider a state $\boldsymbol{s}_t \in \mathcal{S}$, and let $\boldsymbol{s}_{t+1} \in \mathcal{S}$ be obtained from $\boldsymbol{s}_t$ after completing a request that promotes item k from list j to list j', simultaneously demoting i from list j' to list j. The transition rates for RR-c($\boldsymbol{m}$) are then $q(\boldsymbol{s}_t, \boldsymbol{s}_{t+1}) = \sum_{v=1}^{u} \lambda_{vk}(j) c_{vk}(j, j')/m_{j'}$ and $q(\boldsymbol{s}_{t+1}, \boldsymbol{s}_t) = \sum_{v=1}^{u} \lambda_{vi}(j) c_{vi}(j, j')/m_{j'}$. The same holds also for cache misses, with items that reside outside the cache being in list $j = 0$. With the definitions in Theorem 1, we have

$$\pi(\boldsymbol{s}_t) q(\boldsymbol{s}_t, \boldsymbol{s}_{t+1}) = \alpha \gamma_{kj} \gamma_{ij'} \sum_{v=1}^{u} \lambda_{vk}(j) c_{vk}(j, j') \tag{32}$$

where $\alpha = (1/m'_j) \prod_{l \neq j \neq j'} \prod_{k=1}^{m_l} \gamma_{s(k,l)l}$. Similarly,

$$\pi(\boldsymbol{s}_{t+1}) q(\boldsymbol{s}_{t+1}, \boldsymbol{s}_t) = \alpha \gamma_{ij} \gamma_{kj'} \sum_{v=1}^{u} \lambda_{vi}(j) c_{vi}(j, j') \tag{33}$$

If both left-hand sides of (32) and (33) are strictly positive, then they must be equal since $\gamma_{ij'}/\gamma_{ij}$ and $\gamma_{kj'}/\gamma_{kj}$ match the values of the two summations by the definitions in Theorem 1. For the same reasons, if any of the factors in zero, then both left-hand sides of (32) and (33) are zero, e.g., if the summation in (32) is zero, then $\gamma_{kj'}$ in (33) is zero as it also includes this sum in its definition.

Note that the last passages are a consequence of the fact that the paths $\mathcal{P}_k(j')$ and $\mathcal{P}_k(j)$ are stream-independent, as otherwise there could be two streams v and v' such that item k after a request from stream v' reaches list j' from list $l \neq j \neq j'$ and without visiting list j. In this case $c_{v'k}(l, j')\lambda_{v'k}(l)$ would not be present within the summation in (32), whereas it would need to be accounted for to some extent within $\gamma_{kj'}$, as this describes a position for k that is reachable also via the promotion from l.

Applying recursively the last result, it follows that the product-form (1) for any feasible sequence of states $\boldsymbol{s}_0 \rightarrow \boldsymbol{s}_1 \rightarrow \cdots \rightarrow \boldsymbol{s}_S$ satisfies Kolmogorov's criterion [31]

$$\pi(\boldsymbol{s}_0) \prod_{t=1}^{S} q(\boldsymbol{s}_{t-1}, \boldsymbol{s}_t) = \pi(\boldsymbol{s}_S) \prod_{t=1}^{S} q(\boldsymbol{s}_t, \boldsymbol{s}_{t-1})$$

This is a necessary and sufficient condition for the Markov process to be reversible and for (1) to be its equilibrium distribution [31].

2) FIFO-c($\boldsymbol{m}$) policy.: In the case of FIFO-c($\boldsymbol{m}$), the process is not reversible. However, it is sufficient to plug (1) in the global balance equations to verify the statement. The global balance equations are

$$\begin{aligned}
&\left(\sum_{j=0}^{h-1} \sum_{\substack{j'=1..h \\ j' \neq j}} \sum_{k=1}^{m_j} \left(\sum_{v=1}^{u} \lambda_{vs(k,j)}(j) c_{vs(k,j)}(j, j') \right) \right) \pi(\boldsymbol{s}) \\
&= \sum_{j=0}^{h-1} \sum_{\substack{j'=1..h \\ j' \neq j}} \sum_{k=1}^{m_j} \left(\sum_{v=1}^{u} \lambda_{vs(1,j')}(j) c_{vs(1,j')}(j, j') \right) \pi(\boldsymbol{s}_{kjj'})
\end{aligned} \tag{34}$$

where $\boldsymbol{s}$ is reachable from the initial state and where $s \equiv s(i, j)$ and $\boldsymbol{s}$ is obtained from $\boldsymbol{s}_{kjj'} \in \mathcal{S}$ by promoting item $s(1, j')$, assumed in $\boldsymbol{s}_{kjj'}$ at position k of list j, to the front of list j'. We now note that the innermost brackets in the global balance equations if positive may be seen as ratios of cost factors, e.g., if all terms are positive

$$\sum_{j=0}^{h-1} \sum_{\substack{j'=1..h \\ j' \neq j}} \sum_{k=1}^{m_j} \frac{\gamma_{s(k,j)j'}}{\gamma_{s(k,j)j}} \pi(\boldsymbol{s}) = \sum_{j=0}^{h-1} \sum_{\substack{j'=1..h \\ j' \neq j}} \sum_{k=1}^{m_j} \frac{\gamma_{s(1,j')j'}}{\gamma_{s(1,j')j}} \pi(\boldsymbol{s}_{kjj'}) \tag{35}$$

We also observe that the product-form in (1) satisfies

$$\pi(\boldsymbol{s}_{kjj'}) = \frac{\gamma_{s(1,j')j} \gamma_{s(k,j)j'}}{\gamma_{s(1,j')j'} \gamma_{s(k,j)j}} \pi(\boldsymbol{s}) \tag{36}$$

Plugging (36) into (35) the expression reduces to an identity. This holds also if some of the innermost brackets in (34) are zero, since in this case the terms are neglected from the balance equation.

B. Proof of normalizing constant asymptotic expansion

It is possible to verify with a little algebra that

$$\frac{\partial E(\boldsymbol{m})}{\partial \gamma_{kl}} = m_l E_k(\boldsymbol{m} - 1_l) = \frac{\pi_{kl}(\boldsymbol{m})}{\gamma_{kl}} E(\boldsymbol{m}). \tag{37}$$

Applying this relationship to (6) and using (7) we find

$$\pi_{kl}(\boldsymbol{m}) = m_l \gamma_{kl} \frac{E_k(\boldsymbol{m} - 1_l)}{E(\boldsymbol{m})} = \gamma_{kl} \xi_l(\boldsymbol{m}) \frac{E_k(\boldsymbol{m} - 1_l)}{E(\boldsymbol{m} - 1_l)}$$

and the result follows by (4).

C. Proof of normalizing constant asymptotic expansion

1) Derivation of $\phi(\boldsymbol{x}, y)$.: Plugging (12) into (14) we get

$$\begin{aligned}
B(\boldsymbol{x}, y) e^{\phi(\boldsymbol{x},y)/\varepsilon} &= B(\boldsymbol{x}, y - \varepsilon) e^{\phi(\boldsymbol{x},y-\varepsilon)/\varepsilon} \\
&+ \textstyle\sum_{j=1}^{h} g_j(y) B(\boldsymbol{x} - \varepsilon 1_j, y - \varepsilon) e^{\phi(\boldsymbol{x} - \varepsilon 1_j, y - \varepsilon)/\varepsilon}
\end{aligned} \tag{38}$$

Expanding (38) in powers of ε, we obtain at the lowest order $1 = e^{-\phi_y} + \sum_j g_j(y) e^{-\phi_y - \phi_j}$, where ϕ_j (resp. ϕ_y) denotes partial differentiation with respect to x_j (resp. y). Multiplying both sides by e^{ϕ_y} and simplifying yields

$$1 - e^{\phi_y} + \textstyle\sum_{j=1}^{h} g_j(y) e^{-\phi_j} = 0 \tag{39}$$

The method of characteristics for non-linear PDEs [40] yields

$$dy = -e^{\phi_y}dt \tag{40a}$$
$$dx_j = -g_j(y)e^{-\phi_j}dt \tag{40b}$$
$$d\phi = \phi_y dy + \textstyle\sum_j \phi_j dx_j \tag{40c}$$
$$d\phi_y = -\textstyle\sum_j g'(y)e^{-\phi_j}dt \tag{40d}$$
$$d\phi_j = 0 \tag{40e}$$

where all variables are intended as functions of $(t, \boldsymbol{\xi})$, with t being the integration variable over the characteristic curves and let $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_h)$ parameterize the family of curves. To solve this system, we follow a strategy similar to the one used in [19] to analyze normalizing constants in queueing networks; our strategy differs mainly at the boundary condition.

We first solve (40e) as $\phi_j = \log \xi_j$, where we have set the integration constant to $\log \xi_j$, $j = 1, \ldots, h$. Multiplying (40d) by (40a) after dividing both by dt, and integrating we find under the initial conditions $\phi = 0$, $\boldsymbol{x} = 0$, and $y = 0$ if $t = 0$, that $\phi_y = \log(1 + \sum_j g_j(y)\xi_j)$, where intermediate integration constants need to be set to unity for consistency with (39). Plugging the last result in (40a) and integrating we obtain t, after which (40b) yields (16). Using the above results, (40c) integrates to (15).

2) Derivation of $B(\boldsymbol{x}, y)$.: Expanding (38) and matching first-order terms, we write after simplifications $\dfrac{dB}{dt} = \dfrac{B}{2}(\phi_{yy}e^{\phi_y} + \sum_j g_j(y)(\phi_{jj} + 2\phi_{jy})e^{-\phi_j})$. Plugging (15), this becomes similar to the transport equation studied in [19]. Similar passages as in [19] yield $B(\boldsymbol{x}, y) = b(\xi_1, \ldots, \xi_h)(\det \boldsymbol{J})^{-1/2}(1 + \sum_j g_j(y)\xi_j)^{-1/2}$, in which $b(\xi_1, \ldots, \xi_h)$ is an arbitrary function, the matrix $\boldsymbol{J}$ is given in (18), and the ξ_j's depend on $\boldsymbol{x}$ and y via (16).

3) Boundary condition.: At the boundary, we choose to match the arbitrary function $b(\xi_1, \ldots, \xi_h)$ to the value of $H(\boldsymbol{m})$ when $\gamma_{kj} = g_j(k) = \gamma$, $\forall k, j$, for arbitrary γ. In this case, by definition

$$H_0(\boldsymbol{m}, n) = \frac{n!}{(n-m)! m_1! \cdots m_h!} e^{\log \gamma \sum_j j m_j}$$

Using Stirling approximation $n! \sim \sqrt{2\pi n}\left(\dfrac{n}{e}\right)^n$ and scaling the variables in $H_0(\boldsymbol{m}, n)$ to obtain a scaled constant $h_0(\boldsymbol{x}, y)$, we get after simple manipulations

$$h_0(\boldsymbol{x}, y) \sim \frac{(2\pi)^{-h/2}}{\sqrt{\prod_j x_j}} \sqrt{\frac{y}{y-x}} e^{\gamma/\varepsilon}$$

where $\gamma = y \log y - (y - x)\log(y - x) - \sum_j x_j \log x_j + \log \gamma \sum_j j x_j$. For small $\boldsymbol{x}$ and y, we note that (16) gives $x_j = \xi_j \gamma^j y (1 + \sum_{l=1}^h \gamma^l \xi_l)^{-1}$, implying $\xi_j = x_j (y - x)^{-1} \gamma^{-j}$ upon summing over j. Substituting the expression of ξ_j in (15) it is possible to verify that ϕ matches γ. We then require

$$\frac{b(\xi_1, \ldots, \xi_h)}{\sqrt{\det \boldsymbol{J}} \sqrt{1 + \sum_j \gamma^j \xi_j}} = \frac{(2\pi)^{-h/2}}{\sqrt{\prod_j x_j}} \sqrt{\frac{y}{y-x}}$$

By definition of J_{ik}, we find $J_{ik} = (y - x)(\delta_{ik}\gamma^i - \dfrac{x_i}{y}\gamma^k)$. Using this result we compute $\det \boldsymbol{J}$ and determine with a little algebra the required value of $b(\xi_1, \ldots, \xi_h)$, which yields (17). Lastly, to match $H_0(\boldsymbol{m})$, we need to set $D(\varepsilon) = \varepsilon^{h/2}$.

D. Proof of Theorem 3

To prove the lower bound, we show that the assumption implies $\pi_i(\boldsymbol{m}) \geq \pi_k(\boldsymbol{m})$, $\forall \boldsymbol{m}$, so that the statement follows by applying Chebyshev's sum inequality [41] to the denominator of (10). Using (4), we need to show that $E_k(\boldsymbol{m}) \geq E_i(\boldsymbol{m})$ for all i, $k > i$. Note that $E_k(\boldsymbol{m})$ is obtained from $E_i(\boldsymbol{m})$ by replacing γ_{kl} in $E_i(\boldsymbol{m})$ with $\gamma_{il} \geq \gamma_{kl}$. The statement then holds since the normalizing constant is by (37) non-decreasing with the cost factors.

E. Proof of Lemma 1

The proof is by induction on the number of items n and depends on results in Appendix L2, in particular on the recurrence relation

$$\xi_l(\boldsymbol{m}) = \frac{1 + \sum_{\substack{j=1\\j\neq l}}^h \gamma_{kj}\xi_j^{-k}(\boldsymbol{m} - 1_l) + \gamma_{kl}\xi_l^{-k}(\boldsymbol{m} - 1_l)}{\dfrac{I\{n > m\}}{\xi_l^{-k}(\boldsymbol{m})} + \sum_{\substack{j=1\\j\neq l}}^h \gamma_{kj}\dfrac{\xi_j^{-k}(\boldsymbol{m} - 1_l)}{\xi_l^{-k}(\boldsymbol{m} - 1_j)} + \gamma_{kl}} \tag{41}$$

where $I\{n > m\} = 1$ if and only if $n > m$.
We now prove the result.
Case $n = m+1$. The case holds since by (7) it is $\xi_j^{-i}(\boldsymbol{m} - 1_l) = 0$, being $E(\boldsymbol{m} - 1_l - 1_j) = 0$.
Induction step. Let the model with $n - 1$ items exclude item i, we focus on the hypothesis $\xi_j^{-i}(\boldsymbol{m} - 1_l) \leq \xi_j^{-i}(\boldsymbol{m})$, $\forall j, l$. We choose $k = i$ in (41) and divide both sides by $\xi_l^{-i}(\boldsymbol{m})$ so that

$$\frac{\xi_l(\boldsymbol{m})}{\xi_l^{-i}(\boldsymbol{m})} = \frac{1 + \sum_{\substack{j=1\\j\neq l}}^h \gamma_{ij}\xi_j^{-i}(\boldsymbol{m} - 1_l) + \gamma_{il}\xi_l^{-i}(\boldsymbol{m} - 1_l)}{1 + \sum_{\substack{j=1\\j\neq l}}^h \gamma_{ij}\xi_j^{-i}(\boldsymbol{m}) + \gamma_{il}\xi_l^{-i}(\boldsymbol{m})}$$

since $I\{n > m\} = 1$ and by (7) we note that $\xi_j^{-i}(\boldsymbol{m})/\xi_l^{-i}(\boldsymbol{m}) = E_j(\boldsymbol{m} - 1_l)/E_i(\boldsymbol{m} - 1_l) = \xi_j^{-i}(\boldsymbol{m} - 1_l)/\xi_l^{-i}(\boldsymbol{m} - 1_j)$. If we compare term-by-term numerator and denominator in the above expression, by the induction hypothesis we obtain the upper bound $\xi_l(\boldsymbol{m}) \leq \xi_l^{-i}(\boldsymbol{m})$. Plugging (7) in the last inequality, we get $E(\boldsymbol{m} - 1_l)E_i(\boldsymbol{m}) \leq E_i(\boldsymbol{m} - 1_l)E(\boldsymbol{m})$. Diving both sides by $E(\boldsymbol{m} - 1_l)E(\boldsymbol{m})$, by (4) we get $\pi_i(\boldsymbol{m}) \geq \pi_i(\boldsymbol{m} - 1_l)$. The last relationship implies in particular $\pi_i(\boldsymbol{m} - 1_j) \geq \pi_i(\boldsymbol{m} - 1_l - 1_j)$, thus by (10) we find the other bound $\xi_j(\boldsymbol{m} - 1_l) \leq m_j \left(\sum_{i=1}^n \gamma_{il}(1 - \pi_i(\boldsymbol{m} - 1_j))\right)^{-1} = \xi_j(\boldsymbol{m})$.

F. Proof of Lemma 2

We give the proof for $\pi_i(\boldsymbol{m}) = 1 - \pi_{i0}(\boldsymbol{m})$ by showing that $\pi_i(\boldsymbol{m}) \geq \pi_i(\boldsymbol{m} - 1_l)$, for any l. By (4), we need show that $E(\boldsymbol{m} - 1_l)E_i(\boldsymbol{m}) \leq E_i(\boldsymbol{m} - 1_l)E(\boldsymbol{m})$. Plugging (6) in $E(\boldsymbol{m} - 1_l)$ and $E(\boldsymbol{m})$, with the choice $k = i$, after simplifications it is sufficient to show that $E_i(\boldsymbol{m} - 1_l - 1_j)E_i(\boldsymbol{m}) \leq E_i(\boldsymbol{m} - 1_l)E_i(\boldsymbol{m} - 1_j)$, for all $j \neq l$. Dividing by $E_i(\boldsymbol{m})E_i(\boldsymbol{m} - 1_j)$ this may be seen as stating $\xi_j^{-i}(\boldsymbol{m} - 1_l) \leq \xi_j^{-i}(\boldsymbol{m})$, which holds after applying Lemma 1 to a model without item i.

G. Proof of Theorem 4

We obtain the bound by summing (11) over all l, and then by Lemma 2 replacing $\pi_i(\boldsymbol{m} - 1_l)$ with its upper bound $\pi_i(\boldsymbol{m})$. This yields $\pi_i(\boldsymbol{m}) \geq \sum_{l=1}^h \gamma_{il}\xi_l(\boldsymbol{m})(1 - \pi_i(\boldsymbol{m}))$. The proof follows after solving for $\pi_i(\boldsymbol{m})$ and using the result to compute $\pi_{i0}(\boldsymbol{m})$.

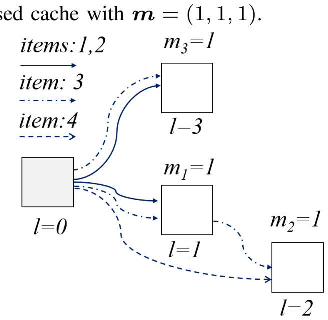
SUPPLEMENTAL MATERIAL

H. Summary of notation

1_j	unit vector in direction j
$c_{vk}(l,j)$	cost (probability) for item k in list l to access list j when requested by stream v
$E(\boldsymbol{m})$	normalizing constant of the equilibrium state probabilities
$E_k(\boldsymbol{m})$	normalizing constant for a model without item k
γ_{kl}	cost factor for item k to access list l
h	number of lists
$\lambda_{vk}(l)$	stream-v request rate for an item k residing in list l
m	cache capacity
$\boldsymbol{m}$	cache capacity vector
m_j	capacity of list j
$M(\boldsymbol{m})$	miss rate
$M_k(\boldsymbol{m})$	item-k miss rate
$M_{vk}(\boldsymbol{m})$	item-k miss rate relatively to requests from stream v
n	number of items
π_{kl}	probability that item k resides in list l
$\boldsymbol{s}$	state vector
$s(i,k)$	index of the item in position i of list k
u	number of streams

I. Example: a cache with a tree topology

Fig. 9. A tree-based cache with $\boldsymbol{m} = (1,1,1)$.



Some transient and recurrent states for a cache with $\boldsymbol{m} = (1,1,1)$ and $n = 4$ items are listed in Table 9. In the cache diagram, an arrow for item i between lists l and l' indicates that there exist a stream v such that $c_{vi}(l,l') > 0$. Transient states that are not reachable according to the item paths determined by the access costs are not listed in the table. The product-form expressions shown for the recurrent states in Table 9 are obtained from (1). We have omitted $\gamma_{10} = \gamma_{20} = \gamma_{30} = 1$ from the probability expressions. In this cache we have, e.g., $\mathcal{P}_3(2) = \{(0,1),(1,2)\}$ and $\mathcal{P}_4(2) = \{(0,2)\}$. Note that according to our definitions $\mathcal{P}_{v1}(3) = \mathcal{P}_{v2}(3) = \mathcal{P}_{v4}(1) = \mathcal{P}_{v4}(2) = \emptyset$ implying that Theorem 1 assumes $\lambda_{v1}(2) = \lambda_{v1}(3) = \lambda_{v4}(1) = \lambda_{v4}(2) = 0$, which influences the summations in the definition of γ_{ij}. A worked example for a possible parameterization of the cache model in Table 9 is given in Appendix K.

TABLE IV
TRANSIENT AND RECURRENT STATES UNDER ITEM-DEPENDENCE COSTS FOR THE CACHE IN FIGURE 9.

state $\boldsymbol{s}$	**type**	**probability $\pi(\boldsymbol{s})$**
$[4,2,1,3]$	transient	0
$[4,1,2,3]$	transient	0
$[3,2,1,4]$	recurrent	$\gamma_{21}\gamma_{12}\gamma_{43}/E(\boldsymbol{m})$
$[3,1,2,4]$	recurrent	$\gamma_{11}\gamma_{22}\gamma_{43}/E(\boldsymbol{m})$
$[2,3,1,4]$	recurrent	$\gamma_{31}\gamma_{12}\gamma_{43}/E(\boldsymbol{m})$
$[2,1,3,4]$	recurrent	$\gamma_{11}\gamma_{32}\gamma_{43}/E(\boldsymbol{m})$
$[1,3,2,4]$	recurrent	$\gamma_{31}\gamma_{22}\gamma_{43}/E(\boldsymbol{m})$
$[1,2,3,4]$	recurrent	$\gamma_{21}\gamma_{32}\gamma_{43}/E(\boldsymbol{m})$

J. Example: a product-form model

We consider again the model in Table 9, in which $n = 4$, $h = 3$, and $\boldsymbol{m} = (1,1,1)$. We consider $u = 2$ streams that both access items at rate $\lambda_{vi}(j)$. If $\mathcal{P}_{vi}(j) = \emptyset$ in which case item i cannot reach list j under requests from stream v and we set $\lambda_{vi}(j) = 0.0$. With these definitions we get

$$\lambda_1 = [\lambda_{1i}(j)] = \begin{bmatrix} 4 & 9 & 6 & 5 \\ 10 & 10 & 6 & 0 \\ 8 & 4 & 9 & 0 \\ 0 & 0 & 3 & 8 \end{bmatrix}$$

$$\lambda_2 = [\lambda_{2i}(j)] = \begin{bmatrix} 8 & 6 & 2 & 10 \\ 3 & 2 & 8 & 0 \\ 2 & 4 & 5 & 0 \\ 0 & 0 & 2 & 7 \end{bmatrix}$$

K. Example: a product-form model

We consider again the model in Table 9, in which $n = 4$, $h = 3$, and $\boldsymbol{m} = (1,1,1)$. We consider $u = 2$ streams that both access items at rate $\lambda_{vi}(j)$. If $\mathcal{P}_{vi}(j) = \emptyset$ in which case item i cannot reach list j under requests from stream v and we set $\lambda_{vi}(j) = 0.0$. With these definitions we get

$$\lambda_1 = [\lambda_{1i}(j)] = \begin{bmatrix} 4 & 9 & 6 & 5 \\ 10 & 10 & 6 & 0 \\ 8 & 4 & 9 & 0 \\ 0 & 0 & 3 & 8 \end{bmatrix}$$

$$\lambda_2 = [\lambda_{2i}(j)] = \begin{bmatrix} 8 & 6 & 2 & 10 \\ 3 & 2 & 8 & 0 \\ 2 & 4 & 5 & 0 \\ 0 & 0 & 2 & 7 \end{bmatrix}$$

Let $C_{vi} = [c_{vi}(l,j)]$ be the access cost matrix for stream v and item i. We assign the following values:

$$C_{11} = \begin{bmatrix} \frac{1}{2} & \frac{1}{6} & \frac{1}{3} & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad C_{12} = \begin{bmatrix} 0 & 1 & \frac{3}{10} & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$C_{13} = \begin{bmatrix} 0 & \frac{7}{10} & \frac{3}{10} & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad C_{14} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$C_{21} = \begin{bmatrix} \frac{2}{3} & \frac{1}{9} & \frac{2}{9} & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad C_{22} = \begin{bmatrix} 0 & 1 & \frac{3}{10} & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$C_{23} = \begin{bmatrix} 0 & \frac{7}{10} & \frac{3}{10} & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad C_{24} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The irreducible Markov process for the recurrent states shown in Table 9 has infinitesimal generator

$$Q = \begin{bmatrix} -8 & 0 & \frac{28}{5} & 0 & 0 & \frac{12}{5} \\ 0 & -8 & 0 & \frac{12}{5} & \frac{28}{5} & 0 \\ 15 & 0 & -\frac{39}{2} & 0 & \frac{9}{2} & 0 \\ 0 & \frac{9}{2} & 0 & -\frac{39}{2} & 0 & 15 \\ 0 & \frac{16}{9} & \frac{32}{9} & 0 & -\frac{16}{3} & 0 \\ \frac{32}{9} & 0 & 0 & \frac{16}{9} & 0 & -\frac{16}{3} \end{bmatrix}$$

For example, $Q(6,1) = \frac{32}{9}$ is the transition rate between state $[1,2,3,4]$ and state $[3,2,1,4]$. We determine numerically from the Markov process the following equilibrium state probabilities

$$\boldsymbol{\pi} = \begin{bmatrix} 0.3635 & 0.0545 & 0.1357 & 0.0291 & 0.1718 & 0.2254 \end{bmatrix}$$

where $\boldsymbol{\pi} = [\pi(\boldsymbol{s})]$. We now verify that this is the same result that one would obtain using the product-form expression. Using the definitions in Theorem 1 we first compute the following cost factors

$$\gamma = [\gamma_{ij}] = \begin{bmatrix} 1 & \frac{16}{9} & \frac{32}{9} & 0 \\ 1 & 15 & \frac{9}{2} & 0 \\ 1 & \frac{28}{5} & \frac{12}{5} & \frac{84}{5} \\ 1 & 0 & 0 & 15 \end{bmatrix}$$

where rows index items $i = 1, \dots, n$ and columns index lists $j = 0, \dots, h$.

With the above parameterization, we obtain from (1) the following state probabilities:

state $\boldsymbol{s}$	**probability $\pi(\boldsymbol{s})$**	**value**
$[3,2,1,4]$	$\gamma_{21}\gamma_{12}\gamma_{43}/E(\boldsymbol{m})$	$\frac{1200}{3301} = 0.3635$
$[3,1,2,4]$	$\gamma_{11}\gamma_{22}\gamma_{43}/E(\boldsymbol{m})$	$\frac{180}{3301} = 0.0545$
$[2,3,1,4]$	$\gamma_{31}\gamma_{12}\gamma_{43}/E(\boldsymbol{m})$	$\frac{448}{3301} = 0.1357$
$[2,1,3,4]$	$\gamma_{11}\gamma_{32}\gamma_{43}/E(\boldsymbol{m})$	$\frac{96}{3301} = 0.0291$
$[1,3,2,4]$	$\gamma_{31}\gamma_{22}\gamma_{43}/E(\boldsymbol{m})$	$\frac{567}{3301} = 0.1718$
$[1,2,3,4]$	$\gamma_{21}\gamma_{32}\gamma_{43}/E(\boldsymbol{m})$	$\frac{810}{3301} = 0.2254$

where we omitted the terms for virtual list 0 since $\gamma_{30} = \gamma_{20} = \gamma_{10} = 1$. The values in the table match the ones obtained numerically for $\boldsymbol{\pi}$.

L. Additional recurrence relations

We develop here additional recurrence relations for the normalizing constant $E(\boldsymbol{m})$ and for the quantities $\xi_l(\boldsymbol{m})$.

1) Recurrence relation for $E(\boldsymbol{m})$.: Plugging (3) into the condition $\sum_i \pi_{il}(\boldsymbol{m}) = m_l$ yields the recurrence relation

$$E(\boldsymbol{m}) = \sum_{i=1}^{n} \gamma_{il} E_i(\boldsymbol{m} - 1_l) \tag{42}$$

for all $l = 1, \dots, h$. The right-hand side may be seen as summing over all the (un-normalized) conditional probabilities that item i is in position m_l in list l. Further, summing (6) over $k = 1, \dots, n$ we get

$$\sum_{k=1}^{n} E(\boldsymbol{m}) = \sum_{k=1}^{n} E_k(\boldsymbol{m}) + \sum_{j=1}^{h} m_j \sum_{k=1}^{n} \gamma_{kj} E_k(\boldsymbol{m} - 1_j)$$

Finally, using (42) on the last summation yields

$$(n - m)E(\boldsymbol{m}) = \sum_{i=1}^{n} E_i(\boldsymbol{m}) \tag{43}$$

2) Recurrence relation for $\xi_l(\boldsymbol{m})$.: We also observe that $\xi_l(\boldsymbol{m})$ admits the exact recurrence relation

$$\xi_l(\boldsymbol{m}) = \frac{1 + \sum_{\substack{j=1 \\ j \neq l}}^{h} \gamma_{kj} \xi_j^{-k}(\boldsymbol{m} - 1_l) + \gamma_{kl} \xi_l^{-k}(\boldsymbol{m} - 1_l)}{\frac{I\{n > m\}}{\xi_l^{-k}(\boldsymbol{m})} + \sum_{\substack{j=1 \\ j \neq l}}^{h} \gamma_{kj} \frac{\xi_j^{-k}(\boldsymbol{m} - 1_l)}{\xi_l^{-k}(\boldsymbol{m} - 1_j)} + \gamma_{kl}} \tag{44}$$

for arbitrary $k = 1, \dots, n$, where I is the indicator function. Relation (44) is a non-uniform cost extension of known results for $F_l(\boldsymbol{m})$ [6, Eq. 2.7] and [15, Thm. 2] in uniform cost models. The proof of (44) follows similarly to earlier work, that is plugging (6) into (7) we get

$$\xi_l(\boldsymbol{m}) = \frac{E_k(\boldsymbol{m} - 1_l) + \sum_{j=1}^{h} (m_j - \delta_{jl}) \gamma_{kj} E_k(\boldsymbol{m} - 1_j - 1_l)}{E_k(\boldsymbol{m}) + \sum_{j=1}^{h} m_j \gamma_{kj} E_k(\boldsymbol{m} - 1_j)} \tag{45}$$

The result is then obtained with simple passages by dividing both numerator and denominator by $E_k(\boldsymbol{m})$ and using that

$$\frac{T_l^{-k}(\boldsymbol{m})}{T_j^{-k}(\boldsymbol{m} - 1_l)} = \frac{E_k(\boldsymbol{m} - 1_j - 1_l)}{E_k(\boldsymbol{m} - 1_l)} \frac{E_k(\boldsymbol{m} - 1_l)}{E_k(\boldsymbol{m})}$$

Using (44), we can compute the miss rate in $\mathcal{O}(hn \prod_{j=1}^{h} (m_h + 1))$. The recursion has boundary conditions: i) $\xi_l(\boldsymbol{m}) = +\infty$ if $m > n$ or $\min_l m_l < 0$; ii) $\xi_l(\boldsymbol{0}) = 0$; iii) $\xi_l(1_l) = \gamma_{k,l}$ if $m_k = 1$ and $n = 1$; iv) $\xi_l(\boldsymbol{m}) = 0$ if $m_l = 0$.

M. Derivation of the refined mean-field approximation (RMF)

In this section, we discuss a refined mean-field approximation, based on the theory developed in [37], for a cache with n items and h lists arranged in a linear topology. We give the derivation for the RR-$c(\boldsymbol{m})$ replacement policy; the equations for FIFO-$c(\boldsymbol{m})$ can be obtained with similar passages. As this derivation is limited to linear topologies, we use a simplified notation. We assume that an item enters the cache in list 1 and moves from list l to list $(l+1)$ at rate $p_{il} = \sum_{v=1}^{u} \lambda_{vi}(l) c_{vi}(l, (l+1))$. Let X_{il} be the indicator that the item i is in list l:

$$X_{il} = \begin{cases} 1 & \text{if and only if item } i \text{ is in list } l \\ 0 & \text{otherwise.} \end{cases}$$

Note that for each list, $l \in \{1 \dots h\}$, we have $\sum_i X_{il} = m_i$ at all times. The transitions for the continuous-time Markov chain underpinning this model are

$$X \mapsto X - e_{il} + e_{i(l+1)} + e_{jl} - e_{j(l+1)} \tag{46}$$

at rate $p_{il} X_{il} X_{j(l+1)}/m_{l+1}$ for $l \in \{1 \dots, h-1\}$,

$$X \mapsto X + e_{i1} - e_{j1} \tag{47}$$

at rate $p_{i0}\left(1 - \sum_l X_{il}\right) X_{j1}/m_1$ for $l = 0$. The first transition corresponds to the item i in list l being exchanged with an item j that is in list $l+1$. The second transition corresponds to an item from outside the cache being exchanged with an item in list 1.

We first derive a standard mean field approximation. It is possible to modify the model described in (46)-(47) into a density dependent population process by scaling the size of the transitions by a factor $1/N$ and accelerating the time by a factor N. The model then becomes

$$X \mapsto X + \frac{1}{N}(-e_{il} + e_{i(l+1)} + e_{jl} - e_{j(l+1)}) \tag{48}$$

at rate $N p_{il} X_{il} X_{j(l+1)}/m_{l+1}$ for $l \in \{1 \dots, h-1\}$.

$$X \mapsto X + \frac{1}{N}(e_{i1} - e_{j1}) \tag{49}$$

at rate $N p_{i0}\left(1 - \sum_l X_{il}\right) X_{j1}/m_1$ for $l = 0$. The case $N = 1$ corresponds to the original model. According to [42], there exists a deterministic dynamical system $x(t) = (x_{il}(t))$ such that for all item i and list l: $X_{il}(t) = x_{il}(t) + O(1/N)$, where x satisfies the differential equation $\dot{x} = f(x)$ with

$$\begin{aligned}
f_{i1}(x) &= p_{i0}\left(1 - \sum_l x_{il}\right)\frac{1}{m_1}\sum_j x_{j1} \\
&\quad - \sum_j p_{j0}\left(1 - \sum_l x_{jl}\right) x_{i1}\frac{1}{m_1} \\
f_{il}(x) &= p_{il-1}x_{il-1}\frac{1}{m_l}\sum_j x_{jl} - p_{il}x_{il}\frac{1}{m_{l+1}}\sum_j x_{j(l+1)} \\
&\quad + \frac{x_{i(l+1)}}{m_{l+1}}\sum_j p_{jl}x_{jl} - \frac{x_{il}}{m_l}\sum_j p_{jl-1}x_{jl-1} \\
f_{ih}(x) &= p_{ih-1}x_{ih-1}\frac{1}{m_h}\sum_j x_{jh} - x_{ih}\frac{1}{m_h}\sum_j p_{jh-1}x_{jh-1}
\end{aligned}$$

where $x \equiv x(t)$ and $x_{il} \equiv x_{il}(t)$. The fixed point of these equations is a global attractor of the above differential equation, see [43].

We are now in a position to derive the refined mean field (RMF) approximation equations. According to [37], [44], there exists a set of time-varying constants $V_{il}(t)$ such that

$$\mathbb{E}[X_{il}(t)] = x_{il}(t) + \frac{1}{N}V_{il}(t) + O(1/N^2),$$

where the above equation holds uniformly in time. The constants V_{il} are obtained via linear ordinary differential equations with time-varying parameters that depend on the first (Jacobian) and second derivative of the drift. Note that in the above equation $\mathbb{E}[X_{il}(t)]$ is equal to the probability that an item i is in list l.

As t goes to infinity, the above expression converges to constant variables x_{il} and V_{il}, where the value $x_{il} = \hat{\pi}_{il}^{FPI}$ is equal to the fixed point approximation given by Equation (24). The number reported in Figure 5 are obtained by evaluating the above expression for $N = 1$.

$$\hat{\pi}_{il}^{RMF} = \hat{\pi}_{il}^{FPI} + V_{il}.$$

1) Derivation of a differential equation for V: Although explicit expressions can be manually obtained from the definitions [37], [44] for such derivatives, it is often more convenient to use symbolic analysis tools such as rmf-tool[2] to determine such expressions. For the model under study, the expressions for the derivatives of the drift are as follows. In what follow, we use a tensor notation to denote the derivatives. For instance $e^{il}_{jl'}$ represents a derivative of the component il with respect to jl' and $e^{il}_{jl',j'l''}$ represents a second order derivative of the component il with respect to jl' and $j'l''$. We use the notation $e^{il}_{jl'}$ to denote a tensor whose $(il)-(jl')$ component is 1 (the others being zero). We also use similar notation for higher order tensors $e^{il}_{jl',j'l''}$ and $e^{il,jl'}$.

Differentiating (46) and (47), the following terms contribute to the first derivative of the drift:

$$-e^{il}_{il} + e^{i(l+1)}_{il} + e^{jl}_{il} - e^{j(l+1)}_{il} \tag{50}$$

at rate $p_{il}X_{j(l+1)}/m_{l+1}$ (for $l \in \{1 \dots, h-1\}$)

$$-e^{il}_{j(l+1)} + e^{i(l+1)}_{j(l+1)} + e^{jl}_{j(l+1)} - e^{j(l+1)}_{j(l+1)} \tag{51}$$

at rate $p_{il}X_{il}/m_{l+1}$ (for $l \in \{1 \dots, h-1\}$),

$$e^{i1}_{il} - e^{j1}_{il} \tag{52}$$

at rate $-p_{i0}X_{j1}/m_1$ (for $l \in \{1 \dots h\}$), and

$$e^{i1}_{j1} - e^{j1}_{j1} \tag{53}$$

at rate $p_{i0}\left(1 - \sum_l X_{il}\right)/m_1$.

Differentiating a second time, we get the following terms:

$$-e^{il}_{il,j(l+1)} + e^{i(l+1)}_{il,j(l+1)} + e^{jl}_{il,j(l+1)} - e^{j(l+1)}_{il,j(l+1)} \tag{54}$$

at rate p_{il}/m_{l+1} (for $l \in \{1 \dots, h-1\}$), and

$$e^{i1}_{il,j1} - e^{j1}_{il,j1} \tag{55}$$

at rate $-p_{i0}/m_1$ for $l \in \{1 \dots h\}$.

Moreover, the matrix Q is the sum of the terms (for $i \neq j$):

$$\begin{aligned}
&e_{il,il} + e_{i(l+1),i(l+1)} + e_{jl,jl} + e_{j(l+1),j(l+1)} \\
&+2(-e_{il,i(l+1)} - e_{il,jl} + e_{il,j(l+1)} \\
&\qquad + e_{i(l+1),jl} - e_{i(l+1),j(l+1)} - e_{jl,j(l+1)})
\end{aligned} \tag{56}$$

at rate $N p_{il}\frac{X_{il}X_{j(l+1)}}{m_{l+1}}$ (for $l \in \{1 \dots, h-1\}$), and

$$e_{i1,i1} + e_{j1,j1} - 2e_{i1,j1} \tag{57}$$

at rate $N p_{i0}(1 - \sum_l X_{il})X_{j1}/m_1$.

[2]https://github.com/ngast/rmf_tool/

From (46), the drift of the model is:

$$
\begin{aligned}
f(x) = \sum_{ijl\in\{1\dots,h-1\}} & (-e_{il} + e_{i(l+1)} + e_{jl} - e_{j(l+1)}) \\
& \cdot p_{il} x_{il} x_{j(l+1)}/m_{l+1} \\
& + \sum_{ij} (e_{i1} - e_{j1}) p_{i0} (1 - \sum_{l} x_{il}) x_{j1}/m_1,
\end{aligned}
$$

where e_{il} is a vector whose (il)-component is 1 (the others being 0).

Let $A(x)$ be the first derivative of the drift. $A(x)$ is a tensor and $A^{il}_{jh}(x) = \partial f_{ij}(x)/\partial x_{jh}$. From (50); A is equal to:

$$
\begin{aligned}
A(x) = \sum_{ij,l\in\{1\dots,h-1\}} & \Bigg[(-e^{il}_{il} + e^{i(l+1)}_{il} + e^{jl}_{il} - e^{j(l+1)}_{il}) \\
& \cdot p_{il} x_{j(l+1)}/m_{l+1} \\
& \qquad + (-e^{il}_{j(l+1)} + e^{i(l+1)}_{j(l+1)} \\
& \cdot + e^{jl}_{j(l+1)} - e^{j(l+1)}_{j(l+1)}) p_{il} x_{il}/m_{l+1} \\
& + \sum_{ij,l\in\{1\dots,h\}} \Bigg[(e^{i1}_{il} - e^{j1}_{il})(-p_{i0} x_{j1}/m_1) \Bigg] \\
& + \sum_{ij} \Bigg[(e^{i1}_{j1} - e^{j1}_{j1}) p_{i0} (1 - \sum_{l} x_{il})/m_1 \Bigg]
\end{aligned}
$$

where e^{il}_{jk} is a tensor whose component $(il),(jk)$ is 1 (the others being zero).

The second derivative $B(x)$ is a tensor whose component $(il),(jl'),(k,l'')$ is $B^{il}_{jl',k,l''}(x) = \partial^2 f_{ij}(x)/(\partial x_{jl'} \partial x_{k,l''})$. From (54), we get:

$$
\begin{aligned}
B(x) = \sum_{ijl\in\{1\dots,h-1\}} & (-e^{il}_{il,j(l+1)} + e^{i(l+1)}_{il,j(l+1)} + e^{jl}_{il,j(l+1)} \\
-e^{j(l+1)}_{il,j(l+1)}) p_{il}/m_{l+1} + & \sum_{ijl\in\{1\dots,h\}} (e^{i1}_{il,j1} - e^{j1}_{il,j1})(-p_{i0}/m_1)
\end{aligned}
$$

Last, the matrix Q is a symmetric positive semi-definite matrix whose component $(il),(jl')$ is $Q_{il,jl'}$. From (56), we get:

$$
\begin{aligned}
Q(x) = \sum_{ijl\in\{1\dots h-1\}} & (e_{il,il} + e_{i(l+1),i(l+1)} + e_{jl,jl} + e_{j(l+1),j(l+1)} \\
& + 2(-e_{il,i(l+1)} - e_{il,jl} + e_{il,j(l+1)} \\
& + e_{i(l+1),jl} \\
& - e_{i(l+1),j(l+1)} - e_{jl,j(l+1)})) p_{il} x_{il} x_{j(l+1)}/m_{l+1} \\
& + \sum_{ij} (e_{i1,i1} + e_{j1,j1} - 2e_{i1,j1}) p_{i0} (1 - \sum_{l} x_{il}) x_{j1}/m_1,
\end{aligned}
$$

Using the above quantities A, B and Q, the refined mean field ODEs are given by:

$$
\begin{aligned}
\frac{d}{dt} W_{il,jl'}(t) &= \sum_{k,l''} (A^{il}_{k,l''}(x(t)) W_{kl'',jl'}(t)) + Q_{il,jl'}(x(t)) \\
\frac{d}{dt} V_{il}(t) &= \sum_{k,l''} A^{il}_{k,l''}(x(t)) V_{k,l'}(t) \\
& \quad + \frac{1}{2} \sum_{jl',k,l''} B^{il}_{jl',kl''}(x(t)) W_{jl',kl''}(t),
\end{aligned}
$$

where $x(t)$ is the solution of the standard mean field approximation. The initial conditions for V and W are 0.

N. Validation: models with synchronous streams

1) Parameters for models with synchronous streams: The following table lists the model parameters for the experiments discussed in Section VIII-C.

n	Number of items	10, 50, 100
n/m	Items-to-capacity ratio	2, 4, 10
h	Number of lists	1, 2, 3
u	Number of streams	2
α_1	Zipf parameter, stream 1	0.6, 1.0, 1.4
α_2	Zipf parameter, stream 2	1.0
σ_1	Mean think time, stream 1	0.001s, 0.01s, 0.1s, 1.0s, 10s
σ_2	Mean think time, stream 2	$2\sigma_1$
θ_{v0}	Mean hit latency	0.01s
θ_{v1}	Mean miss latency	0.10s
s_v	Maximum outstanding requests per stream	1
c	Normalized access cost	1.0

2) Sensitivity to number of outstanding requests.: We have also considered variants where for every stream v the maximum number of outstanding requests is varied as $s_v = 1, 4, 16$, and different mixes of think time values are used in the model. We keep the number of samples used in the simulations to 10^7. In these experiments, we set $n = 10, 50, 100$, $h = 2$, $c = 0.5$, $\alpha_1 = 1.4$, and use both FIFO-$c(\boldsymbol{m})$ and RR-$c(\boldsymbol{m})$, for a total of 162 experiments. The results indicate that average errors remain small for all mixes (σ_1, σ_2), with FPI ranging between 2.01% and 2.33% mean absolute percentage errors across the mixes and with SPA between 0.18% and 0.49%. For both methods, the largest error for FPI is observed for mix $(16, 1)$, while for SPA is for mix $(1, 16)$.

3) Sensitivity to access costs and tree structure.: We now consider another variant where the cache has $h = 5$ lists arranged in a tree structure. As before, we consider $n = 10, 50, 100$, $n/m = 10, 50, 100$, $\alpha_1 = 0.6, 1.0, 1.4$, $\alpha_2 = 1.0$, and $s_1 = s_2 = 1$. Access costs for list $l = 0$, corresponding to cache misses, are set to 1.0, for all $v = 1, \dots, u$. Access costs for stream v requests to item k with a cache hit are given by the following cost matrix

$$
C_{vk} = \begin{bmatrix}
0 & 1 & & & & \\
 & \frac{k\cdot v}{1+k\cdot v} & \frac{1}{3(1+k\cdot v)} & & \frac{2}{3(1+k\cdot v)} & \\
 & & \frac{v}{1+v} & \frac{1}{1+v} & & \\
 & & & 1 & & \\
 & & & & \frac{3}{10} & \frac{7}{10} \\
 & & & & & 1
\end{bmatrix}
$$

where element in row l and column j corresponds to $c_{vk}(l, j)$, for $l, j = 0, \dots, h$. Costs on the main diagonal give the probability that the item upon a hit does not get promoted to another list.

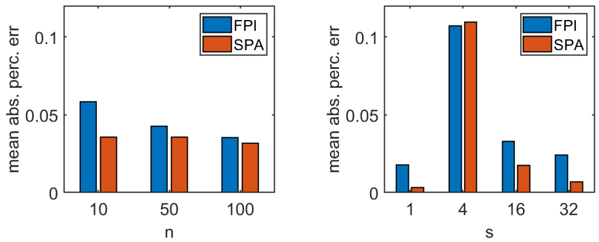


Fig. 10. FPI and SPA item miss ratios on the parallel topology in Figure 7 with correlated item reads.

n	Number of items	10, 50, 100
n/m	Items-to-capacity ratio	2, 4, 10
h	Number of lists	2
u	Number of streams	2
α_{c1}	Zipf parameter at cache c, stream 1	1.4
α_{c2}	Zipf parameter at cache c, stream 2	1.0
θ_{t1}	Mean think time, stream 1	1.0s
θ_{t2}	Mean think time, stream 2	2.0s
θ_{h1}	Mean hit latency, stream 1	0.01s
θ_{h2}	Mean hit latency, stream 2	0.01s
θ_{m1}	Mean miss latency, stream 1	0.10s
θ_{m2}	Mean miss latency, stream 2	0.50s
s	Maximum outstanding requests in each stream	1, 4, 16, 32
c	Normalized access cost	1.0

With the above parameters we find that for FIFO-c($\boldsymbol{m}$) the average errors are 1.23% for FPI and 0.38% for SPA, while maximum errors are 4.34% for FPI and 1.41% for SPA. We also repeat the experiment with RR-c($\boldsymbol{m}$) finding average errors of 1.21% for FPI and 0.32% for SPA, and maximum errors of 4.17% for FPI and 1.41% for SPA, which are very close to the FIFO-c($\boldsymbol{m}$) results and compatible with simulation error.

O. Validation: models with queueing

1) Parameters for models with queueing: The following table lists the model parameters for the experiments discussed in Section IX-C. We index nodes as t (think time), c (cache), h (hit queue), and m (miss queue).

2) Correlated item reads.: We consider a variant of the parallel topology in Figure 7 in which a request that completes in stream 1 deterministically switches class at the reference station (think time node) so to re-enter in the following cycle as a synchronous request of stream 2, and vice-versa for requests completing in stream 2. This is sufficient to introduce a degree of serial correlation in the item references at the cache, since each requests will deterministically alternate requests drawn for Zipf-like distributions with different parameters ($\alpha_{c1} \neq \alpha_{c2}$).

While our approximations cannot explicitly represent correlations, the results shown in Figure 10 show that both FPI and SPI remain fairly accurate on the considered model, especially with small and large number of outstanding requests. This suggests that the proposed methods can still provide reasonably accurate results outside the assumptions.