On Resource Pooling and Separation for LRU Caching

JIAN TAN, GUOCONG QUAN, KAIYI JI, and NESS SHROFF, The Ohio State University, USA

Caching systems using the Least Recently Used (LRU) principle have now become ubiquitous. A fundamental question for these systems is whether the cache space should be pooled together or divided to serve multiple flows of data item requests in order to minimize the miss probabilities. In this paper, we show that there is no straight yes or no answer to this question, depending on complex combinations of critical factors, including, e.g., request rates, overlapped data items across different request flows, data item popularities and their sizes. To this end, we characterize the performance of multiple flows of data item requests under resource pooling and separation for LRU caching when the cache size is large.

Analytically, we show that it is asymptotically optimal to jointly serve multiple flows if their data item sizes and popularity distributions are similar and their arrival rates do not differ significantly; the self-organizing property of LRU caching automatically optimizes the resource allocation among them asymptotically. Otherwise, separating these flows could be better, e.g., when data sizes vary significantly. We also quantify critical points beyond which resource pooling is better than separation for each of the flows when the overlapped data items exceed certain levels. Technically, for a broad class of heavy-tailed distributions we derive the asymptotic miss probabilities of multiple flows of requests with varying data item sizes in a shared LRU cache space. It also validates the characteristic time approximation under certain conditions. These results provide new insights on improving the performance of caching systems.

CCS Concepts: • **Mathematics of computing** → *Queueing theory*; *Stochastic processes*; • **General and reference** → *Metrics*;

Additional Key Words and Phrases: Caching algorithm; LRU; Miss probability; Memcached

ACM Reference Format:
Jian Tan, Guocong Quan, Kaiyi Ji, and Ness Shroff. 2018. On Resource Pooling and Separation for LRU Caching. *Proc. ACM Meas. Anal. Comput. Syst.* 2, 1, Article 5 (March 2018), 31 pages. https://doi.org/10.1145/3179408

1 INTRODUCTION

Caching systems using the Least Recently Used (LRU) principle are already widely deployed but need to efficiently scale to support emerging data applications. They have very different stochastic dynamics [17, 18, 28, 38, 40, 45, 46, 49, 61, 76, 82] than well-studied queueing systems. One cannot apply the typical intuition of resource pooling for queueing, e.g., [15, 23, 52, 62, 80], to caching. To serve multiple flows of data item requests, a fundamental question is whether the cache space should be pooled together or divided (see Fig. 1) in order to minimize the miss probabilities (a.k.a. miss ratios).

This work was supported by the National Science Foundation under Grant No. 1717060 and No. 1719371.

Authors' address: Jian Tan, tan.252@osu.edu; Guocong Quan, quan.72@osu.edu; Kaiyi Ji, ji.367@osu.edu; Ness Shroff, shroff.11@osu.edu, The Ohio State University, 2015 Neil Ave, Columbus, OH, 43210, USA.

2476-1249/2018/3-ART5 $15.00
https://doi.org/10.1145/3179408

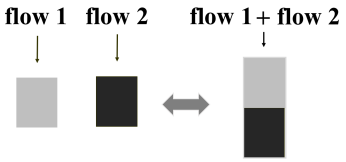


Fig. 1. Flows served separately and jointly

A request is said to "miss" if the corresponding data item is not found in the cache; otherwise a "hit" occurs. For a web service each miss often incurs subsequent work at a backend database, resulting in overhead as high as a few milliseconds or even seconds [86]. A study on Facebook's memcached workloads shows that a small percentage of miss ratio on one server can trigger millions of requests to the database per day [9, 88]. Thus, even a minor increase in the hit ratio can significantly improve system performance. To further motivate the problem, we examine the cache space allocation for in-memory key-value storage systems.

1.1 Background and current practice

In-memory cache processing can greatly expedite data retrieval, since data are kept in Random Access Memory (RAM). In a typical key-value cache system, e.g., Memcached [1, 39], a data item is added to the cache after a client has requested it and failed. When the cache is full, an old data item has to be evicted to make room for the new one. This selection is determined by the caching algorithm. Different caching algorithms have been proposed [60, 66]. However, due to the cost of tracking access history, often only LRU or its approximations [83] are adopted [9]. The LRU algorithm replaces the data item that has not been used for the longest period of time.

The current engineering practice is to organize servers into pools based on applications and data domains [9, 27, 69]. On a server, the cache space is divided into isolated slabs according to data item sizes [1, 88]. Note that different servers and slabs have separate LRU lists. These solutions have yielded good performance [1, 31, 88], through coarse level control on resource pooling and separation. However, it is not clear whether these rules of thumb are optimal allocations, or whether one can develop simple solutions to further improve the performance.

1.2 The optimal strategy puzzle

These facts present a dilemma. On the one hand, multiple request flows benefit from resource pooling. For example, a shared cache space that provides sufficiently high hit ratios for two flows can improve the utilization of the limited RAM space, especially when the two flows contain overlapped common data items so that a data item brought into cache by one flow can be directly used by the other. On the other hand, resource separation facilitates capacity planning for different flows and ensures adequate quality of service for each. For example, a dedicated cache space can prevent one flow with a high request rate from evicting too many data items of another competing flow on the same cache [9].

This dilemma only scratches the surface of whether resource pooling or separation is better for caching. Four critical factors complicate the problem and jointly impact the cache miss probabilities, including request rates, overlapped data items across different request flows, data item popularities and their sizes. Depending on the setting, they may lead to different conclusions. Below we demonstrate the complexity of the optimal strategy using three examples, showing that resource pooling can be asymptotically equal to, better or worse than separation, respectively. Consider two independent flows (1 and 2) of requests with Poisson arrivals of rates ν_1 and ν_2, respectively. The data items of the two flows do not overlap and have unit sizes unless explicitly specified. Their

popularities follow truncated Zipf's distributions, $p_i^{(1)} = c_1/i^{\alpha_1}$ and $p_j^{(2)} = c_2/j^{\alpha_2}, 1 \le i, j \le N$, where i, j are the indeces of the data items of flow 1 and 2, respectively. For pooling, two flows share the whole cache. For separation, the cache is partitioned into two parts using fractions u_1 and u_2, to serve flow 1 and 2 separately, $u_1 + u_2 = 1, u_1, u_2 \ge 0$.

Case 1: Asymptotic equivalence

The optimal resource separation scheme has recently been shown to be better than pooling [29] under certain assumptions based on the characteristic time approximation [28, 38]. However, it is not clear whether the difference is significant or not, especially when the cache size is large (a typical scenario). The first example shows that they can be quite close. Notably resource pooling is self-organizing and need not optimize separation fractions u_1. For $\alpha_1 = 1.5, \alpha_2 = 4.0, \nu_1 = 0.1, \nu_2 = 0.9, N = 10^6$, we plot the overall miss ratios under resource pooling and the optimal separation in Fig. 2, respectively. The optimal ratio u_1 for separation is obtained numerically by an exhaustive search. When the cache size is small, the optimal separation strategy achieves a lower miss ratio

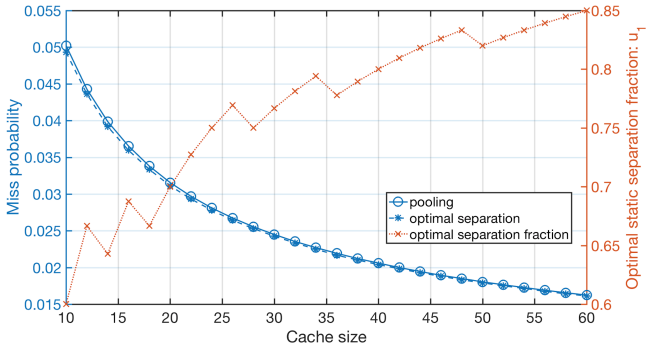


Fig. 2. Asymptotically equal miss ratios

than resource pooling. However, for large cache sizes, the miss ratios are indistinguishable. This is not an coincidence, as shown by Theorem 4.1. Note that the cache sizes take integer values, thus u_1 varying up and down.

Case 2: Pooling is better

The previous example shows that resource pooling can adaptively achieve the best separation fraction when the cache space is large. Consider two flows with $\alpha_1 = \alpha_2 = 2, N = 10^6$ and time-varying Poisson request rates. For $T = 10^6$, let $\nu_1 = 0.1, \nu_2 = 0.9$ in the time interval $[2kT, (2k+1)T)$ and $\nu_1 = 0.9, \nu_2 = 0.1$ in $[(2k+1)T, (2k+2)T)$, $k = 0, 1, 2, \cdots$. The simulation results in Fig. 3

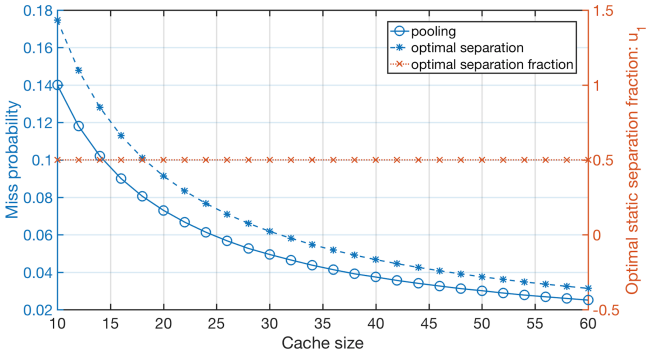


Fig. 3. Benefits of pooling due to self-organization

show that resource pooling yields a smaller miss probability, which primarily attributes to the self-organizing property characterized by Theorem 4.1. Specifically, resource pooling can adaptively achieve the optimal overall miss ratio after an adjustment period in each interval $[kT, (k+1)T)$, $k = 0, 1, 2, \cdots$, while a static separation method cannot always be optimal. The optimal static separation ratio in this case is $u_1 = 0.5$ due to symmetry.

Case 3: Separation is better

Assume that the data items from flow 1 and flow 2 have different sizes 1 and 4, respectively, with $N = 10^6, \alpha_1 = \alpha_2 = 2, \nu_1 = \nu_2 = 0.5$. The simulation results in Fig. 4 show that the optimal separation yields a better performance due to varying data item sizes, which is supported by Theorem 4.1. This may explain why in practice it is beneficial to separate cache space according to applications, e.g., text and image objects, which could have significantly different item sizes [69, 87]. What if the data item sizes are equal? Fig. 2 is an example that separation is better when the cache space is small even with equal data item sizes. However, a small cache may not be typical for caching systems.

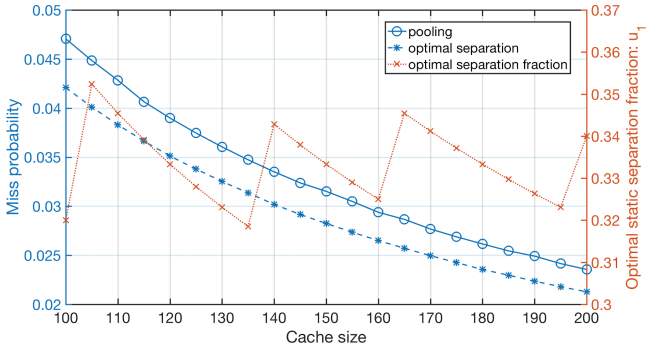


Fig. 4. Benefits of separation due to isolation

These examples motivate us to systematically study the miss probabilities for competing flows with different rates, distributions, and partially overlapped data items of varying sizes. Our analytical results can be used to explain the puzzling performance differences demonstrated in the previous three examples.

1.3 Summary of contributions

(1) An analytical framework under the independent reference model (IRM) [32] is proposed to address four critical factors for LRU caching: request rates, distributions, data item sizes and the overlapped data items across different flows. We derive the asymptotic miss probabilities of multiple flows of requests with varying data item sizes in a shared LRU cache space for a broad class of heavy-tailed distributions, including regularly varying distributions and heavy-tailed Weibull. These asymptotic results validate the characteristic time approximation [28, 38] under certain conditions.

(2) Based on the miss probabilities for both the aggregated and the individual flows, we provide guidance on whether multiple competing flows should be served together or not. First, we show that when the flows have similar distributions and equal data item sizes, the self-organizing property of LRU can adaptively search for the optimal resource allocation for shared flows. As a result, the overall miss probability of the aggregated flows is asymptotically equal to the miss probability using the optimal static separation scheme. In addition, if the request rates of these flows are close, the miss probabilities of individual flows when served jointly differ only by a small constant factor compared to the case when they are served separately. Otherwise, either some of the request

flows will be severely penalized or the total miss ratio will become worse. In that case, it is better to separately serve them. Second, we consider multiple flows with overlapped data. When the overlapped data items exceed a certain level, there exists a region such that every flow can get a better hit ratio. However, if not in this region, e.g., when the arrival rates are very different, some flows will be negatively impacted by other competing flows. Based on the analysis, we discuss engineering implications.
(3) Extensive simulations are conducted to verify the theoretical results. We design a number of simulations, with different purposes and emphases, and show an accurate match with theoretical results.

1.4 Related work

LRU caching is a self-organizing list [2, 3, 5, 21, 22, 37, 51, 53, 73] that has been extensively studied. There are two basic approaches to conduct the analysis: combinatorial and probabilistic. The first approach focuses on the classic amortized [16, 24, 75, 78, 79] and competitive analysis [8, 25, 30, 55, 63]. The second approach includes average case analysis [4, 67, 77] and stochastic analysis [14, 35, 41–43, 68]. When cache sizes are small, the miss probabilities can be explicitly computed [10–12, 50]. For large cache sizes, a number of works (e.g., [17, 44, 48, 61, 74]) rely on the characteristic time approximation [28, 38], which has been extended to cache networks [17, 44, 47, 48, 64, 76]. For fluid limits as scaling factors go to infinity (large cache sizes), mean field approximations of the miss probabilities have been developed [49, 54, 84]. For emerging data processing systems, e.g., Memcached [1], since the cache sizes are usually large and the miss probabilities are controlled to be small, it is natural to conduct the asymptotic analysis of the miss probabilities [56–59]. Although the miss ratios are small, they still significantly impact the caching system performance. Nevertheless, most existing works do not address multiple competing request flows on a shared cache space, which can impact each other through complicated ways.

Workload measurements for caching systems [6, 7, 9, 26, 33, 36, 65] are the basis for theoretical modeling and system optimization. Empirical trace studies show that many characteristics of Web caches can be modeled using power-law distributions [7, 89], including, e.g., the overall data item popularity rank, the document sizes, the distribution of user requests for documents [6, 13, 26, 65], and the write traffic [89]. Similar phenomena have also been found for large-scale key-value stores [9]. These facts motivate us to exploit the heavy-tailed workload characteristics.

Web and network caching is closely related to this study with a large body of dedicated works; see the surveys [71, 85] and the references therein. Recently a utility optimization approach [29, 34] based on the characteristic time approximation [17, 28, 38] has been used to study cache sharing and partitioning. It has concluded that under certain settings the optimal resource separation is better than pooling. However, it is not clear whether the difference is significant or not, especially when the cache size is large for a typical scenario. We show that a simple LRU pooling is asymptotically equivalent to the optimal separation scheme under certain settings, which is significant since the former is adaptive and does not require any configuration or tuning optimization. We focus on the asymptotic miss probabilities for multiple competing flows, as the miss ratio is one of the most important metrics for caching systems with large cache sizes in practice.

2 MODEL AND INTUITIVE RESULTS

Consider M flows of i.i.d. random data item requests that are mutually independent. Assume that the arrivals of flow k follow a Poisson process with rate $\lambda_k > 0, 1 \le k \le M$. The arrivals of the mixed M request flows occur at time points $\{\tau_n, -\infty < n < +\infty\}$. Let I_n be the index of the flow for the request at τ_n. The event $\{I_n = k\}$ represents that the request at τ_n originates from flow k. Due to the Poisson assumption, we have $\mathbb{P}[I_n = k] = \lambda_k / (\sum_i \lambda_i)$.

To model the typical scenario that the number of distinct data items far exceeds the cache size, we assume that each flow can access an infinite number of data items. Formally, flow k accesses the set of data items $d_i^{(k)}, i = 1, 2, \cdots, \infty, 1 \le k \le M$, from which only a finite number can be stored in cache due to the limited capacity. Let $s_i^{(k)}$ denote the size of data item $d_i^{(k)}$. Note that it is possible, and even common in practice, to observe $d_i^{(k)} \equiv d_j^{(g)}$ for flows k and g, where "$\equiv$" means that the two involved data items are the same. Therefore, this model describes the situation when data items can overlap between different flows. For example, in Fig. 5, we have $d_4^{(1)} \equiv d_2^{(2)}$, $d_5^{(1)} \equiv d_3^{(2)}$ and

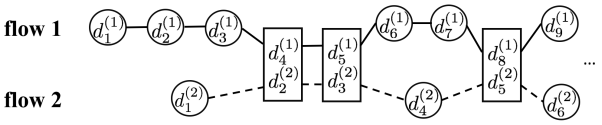


Fig. 5. Data items overlap between two flows

$d_8^{(1)} \equiv d_5^{(2)}$. Let R_n denote the requested data item at time τ_n. Thus, the event $\{I_n = k, R_n = d_i^{(k)}\}$ means that the request at time τ_n is from flow k to fetch data item $d_i^{(k)}$. We also abuse the notation for R_n a bit and define $\mathbb{P}[R_0 > x \mid I_0 = k]$ to be the probability that the request at time τ_0 is to fetch a data item with an index larger than x in the ordered list $\left(d_i^{(k)}, i = 1, 2, 3, \cdots\right)$ of flow k. The ordering will be specified in the following part.

When the system reaches stationarity (Theorem 1 of [56]), the miss ratio of the system is equal to the probability that a request R_0 at time $\tau_0 = 0$ finds that its asked data item is not kept in the cache. Therefore, we only need to consider R_0 in the following part. Due to multiple request flows, we have two sets of probabilities for each flow. Flow k experiences the unconditional probabilities

$$\mathbb{P}\left[R_0 = d_i^{(k)}\right] = p_i^{(k)}, i = 1, 2, 3, \cdots \tag{1}$$

and the conditional probabilities

$$\mathbb{P}\left[R_0 = d_i^{(k)} \middle| I_0 = k\right] = q_i^{(k)}, i = 1, 2, 3, \cdots \tag{2}$$

In general, $q_i^{(k)}$ can be very different from $p_i^{(k)}$, since the multiple request flows not only access distinct data items, but also share common data items, as shown in Fig. 5. Note that we have

$$p_i^{(k)} = \sum_{j=1}^{M} \nu_j \mathbb{P}\left[R_0 = d_i^{(k)} \middle| I_0 = j\right], \tag{3}$$

where $\nu_j \triangleq \mathbb{P}[I_0 = j]$. When there are no overlapped data items across different flows, we have $p_i^{(k)} = \nu_k q_i^{(k)}$. Specially, if there is only a single flow k, i.e., $\mathbb{P}[I_0 = k] = 1$, then $q_i^{(k)} = p_i^{(k)}$ for all i. For multiple request flows, they are coupled through (3), since a data item requested by flow k is more likely to be found in the cache when it has recently been requested by other flows. In this case, the usual belief is to pool these flows together, so that one flow can help the others to increase the hit. However, if the fraction of overlapped data items is not significant enough, it is intuitively inevitable that the help obtained from other flows on these common data items will be limited. There have been no analytical studies to quantify the effects on how the overlapped data can help different flows.

When studying flow k, assume that the data items $d_i^{(k)}$ are sorted such that the sequence $p_i^{(k)}$ is non-increasing with respect to i. Given (3), the sequence $q_i^{(k)}$ is not necessarily non-increasing by this ordering. The miss ratio depends on the popularities $q_i^{(k)}$ and $p_i^{(k)}$, $i \ge 1$, $1 \le k \le M$.

Interestingly, it can be characterized by the following functional relationship $\Phi_k(\cdot)$ for flow k, $1 \le k \le M$, in a neighborhood of infinity,

$$\left(\sum_{i=y}^{\infty} q_i^{(k)}\right)^{-1} \sim \Phi_k\left(\left(p_y^{(k)}\right)^{-1}\right),\ y \to \infty. \tag{4}$$

Note $f(x) \sim g(x)$ means $\lim_{x\to\infty} f(x)/g(x) = 1$. The values in (4) are defined using reciprocals, as both $\left(\sum_{i=y}^{\infty} q_i^{(k)}\right)^{-1}$ and $\left(p_y^{(k)}\right)^{-1}$ take values in $[1, \infty)$, in line with the condition that $\Phi_k(\cdot)$ is defined in a neighborhood of infinity. In the proof of Theorem 3.1, we show that this functional relationship can be leveraged to characterize the asymptotic miss ratios explicitly. We consider the following class of heavy-tailed distributions

$$\lim_{n\to\infty} q_n^{(k)}/q_{n+1}^{(k)} = 1. \tag{5}$$

It includes heavy-tailed Weibull distributions $q_n^{(k)} \sim d\exp(-cn^{\alpha})$, $c, d > 0, 0 < \alpha < 1$, and Zipf's distributions $q_n^{(k)} \sim c/n^{\alpha}$, $c, \alpha > 0$.

It has been shown [41, 43, 56, 59] that the miss probability of LRU is equivalent to the tail of the searching cost distribution under move-to-front (MTF). For MTF, the data items are sorted in increasing order of their last access times. Each time a request is made for a data item, this data item is moved to the first position of the list and all the other data items that were before this one increase their positions in the list by one.

Definition 2.1. Define C_n to be the summation of the sizes for all the data items in the sorted list under MTF that are in front of the position of the data item requested by R_n at time τ_n.

If the cache size is x, then a cache miss under MTF, which is equivalent for LRU policy, can be denoted by $\{C_n > x\}$. For a special case when the data item sizes satisfy $s_i^{(k)} \equiv 1$ for all k, i, the event $\{C_n > x\}$ means the position of the data item in the list is larger than x under MTF.

For the M flows mixed together, let $\{d_i, i = 1, 2, \cdots\}$ denote the set of data items requested by the entirety of these flows, with $\mathbb{P}[R_0 = d_i] = p_i^{\circ}$. Let s_i denote the size of data item d_i and assume $\bar{s} \triangleq \sup_i s_i < \infty$. In general, s_i can take different values when data item sizes vary. Let $M(n)$ denote the total size of all the distinct data items that have been requested on points $\{\tau_{-1}, \tau_{-2}, \cdots, \tau_{-n}\}$. Define

$$m(z) = \sum_{i=0}^{\infty} s_i \left(1 - (1 - p_i^{\circ})^z\right). \tag{6}$$

We have $m(n) = \mathbb{E}[M(n)]$. Since $m(z)$ is strictly increasing, its inverse function $m^{\leftarrow}(z)$ exists and is related to the characteristic time approximation [28, 38]. We can analytically derive $m^{\leftarrow}(z)$ in some typical cases, as shown in Corollaries 3.3 and 3.5, which directly exploit the properties of the popularity distributions, different from the characteristic time approximation.

One of our main results can be informally stated as follows; the rigorous version is presented in Theorem 3.1. Recall a gamma function $\Gamma(\beta_k + 1) = \int_0^{\infty} y^{\beta_k} e^{-y} dy$.

Miss ratio (informal description) *For M flows sharing a cache, if $\Phi_k(x)$, $1 \le k \le M$, is approximately a polynomial function ($\approx x^{\beta_k}$), then, under mild conditions, we obtain, when the cache size x is large enough,*

$$P[\textit{miss ratio of flow } k] \approx \frac{\Gamma(\beta_k + 1)}{\Phi_k(m^{\leftarrow}(x))}. \tag{7}$$

Based on this result, we derive the following **engineering implications** for M flows of data item requests with $q_i^{(k)} \sim c_k/i^{\alpha_k}$, $\alpha_k \geq 1$, $1 \leq k \leq M$.

- **Asymptotically equal**: For M flows of disjoint requests with identical item sizes, the overall miss ratio achieved by resource pooling is asymptotically equal to that achieved by the optimal separation method. This is referred to as the self-organizing behavior of resource pooling (see Section 4.1).
- **Separation is better**: For M flows of disjoint requests with various item sizes, the optimal resource separation achieves a lower overall miss ratio than resource pooling; see Section 4.1.
- **Pooling is better**: For M flows of disjoint requests with identical item sizes, if the statistics, such as request rates, are time-varying, a static separation method cannot always achieve the optimal overall miss ratio. However, due to the self-organizing behavior, resource pooling adaptively adjusts to the optimal solution, as long as the statistics are relatively stable before it converges. Moreover, if there are overlapped data items across these flows, resource pooling can achieve a lower overall miss ratio than the optimal separation. Particularly, the asymptotic miss ratio of each individual flow can be lower under resource pooling, if certain parameters are in a good region as shown in Section 4.3.

Sketch of the proof: First, we derive a representation for the miss probability of the request R_0. Similar arguments have been used in [46, 56] but we take a different approach. Among all the requests that occur before $\tau_0 = 0$ we find the last one that also requests data item R_0. More formally, define $-\sigma$ to be the largest index of the request arrival before τ_0 such that $R_{-\sigma} = R_0$. Conditional on $\{R_0 = d_i^{(k)}\} \cap \{I_0 = k\}$, the following requests $R_{-1}, R_{-2}, R_{-3}, \cdots$ are i.i.d, satisfying $\mathbb{P}\left[R_{-j} = d_i^{(k)} | \{R_0 = d_i^{(k)}\} \cap \{I_0 = k\}\right] = p_i^{(k)}, j \geq 1$, which implies

$$\mathbb{P}\left[\sigma > n \middle| \{R_0 = d_i^{(k)}\} \cap \{I_0 = k\}\right] = \left(1 - p_i^{(k)}\right)^n.$$

Thus, unconditional on R_0, we obtain, recalling (1) and (2),

$$\mathbb{P}\left[\sigma > n \middle| I_0 = k\right] = \sum_{i=1}^{\infty} q_i^{(k)} \left(1 - p_i^{(k)}\right)^n. \tag{8}$$

Now we argue that the event $\{C_0 > x\}$ is completely determined by the requests at the time points $\{\tau_{-1}, \tau_{-2}, \cdots, \tau_{-\sigma}\}$. Recall that $M(n)$ is the total size of all the distinct data items that have been requested on points $\{\tau_{-1}, \tau_{-2}, \cdots, \tau_{-n}\}$. Define the inverse function of $M(n)$ to be $M^{\leftarrow}(x) = \min\{n : M(n) \geq x\}$. We claim that

$$\{C_0 > x\} = \{\sigma > M^{\leftarrow}(x)\}. \tag{9}$$

If the event $\{\sigma > M^{\leftarrow}(x)\}$ happens, the total size of the distinct data items requested on the time interval $(\tau_{-\sigma}, 0)$ is no smaller than x and these data items are different from the one that is requested at time τ_0 (or $\tau_{-\sigma}$). Due to the equivalence of LRU and MTF, when R_0 arrives at τ_0, all of the data items requested on $(\tau_{-\sigma}, 0)$ will be listed in front of it under MTF. Combining these two facts we obtain $\{\sigma > M^{\leftarrow}(x)\} \subseteq \{C_0 > x\}$. If $\{C_0 > x\}$ occurs, then after $\tau_{-\sigma}$ when R_0 is listed in the first position of the list, there must be enough distinct data items that have been requested on $(\tau_{-\sigma}, 0)$ so that their total size exceeds or reaches x. This yields $\{C_0 > x\} \subseteq \{\sigma > M^{\leftarrow}(x)\}$, which proves (9) and implies

$$\mathbb{P}[C_0 > x | I_0 = k] = \mathbb{P}[\sigma > M^{\leftarrow}(x) | I_0 = k]. \tag{10}$$

In order to compute $\mathbb{P}[\sigma > M^{\leftarrow}(x) | I_0 = k]$, we take two steps. The first step is to show

$$\mathbb{P}[\sigma > n | I_0 = k] \approx \Gamma(\beta_k + 1)/\Phi_k(n).$$

The second step is to relate $M^{\leftarrow}(x)$ to $m^{\leftarrow}(x)$ as $x \to \infty$.

Here, we provide an intuitive proof for $\beta_k > 0$. From (8), we have

$$\mathbb{P}\left[\sigma > n | I_0 = k\right] = \sum_{i=1}^{\infty} q_i^{(k)} (1 - p_i^{(k)})^n \approx \sum_{i=1}^{\infty} q_i^{(k)} e^{-np_i^{(k)}}. \tag{11}$$

In order to approximate the summation (11) and obtain an explicit expression, we use the functional relationship $\Phi_k(\cdot)$ for flow k introduced in (4). Combining (4) and (11), we have, for $Q_i = \sum_{j=i}^{\infty} q_j^{(k)}$,

$$\begin{aligned}\mathbb{P}\left[\sigma > n | I_0 = k\right] &\approx \sum_{i=1}^{\infty} q_i^{(k)} e^{-n \Big/ \Phi_k^{\leftarrow}\left(\left(\sum_{j=i}^{\infty} q_j^{(k)}\right)^{-1}\right)} = \sum_{i=1}^{\infty} (Q_i - Q_{i+1}) e^{-n \Big/ \Phi_k^{\leftarrow}(Q_i^{-1})} \\ &\approx \int_0^1 e^{-n \Big/ \Phi_k^{\leftarrow}(x^{-1})} dx \approx \Gamma(\beta_k + 1)/\Phi_k(n). \end{aligned} \tag{12}$$

For the second step, we have $M(n) \approx \mathbb{E}[M(n)] = m(n)$ with a high probability as $n \to \infty$ by a concentration inequality. The monotonicity and continuity of $m(n)$ imply $M^{\leftarrow}(x) \approx m^{\leftarrow}(x)$ with a high probability under certain conditions. Applying (10) and (12), we finish the proof

$$\begin{aligned}\mathbb{P}[C_0 > x | I_0 = k] &= \mathbb{P}[\sigma > M^{\leftarrow}(x) | I_0 = k] \\ &\approx \mathbb{P}[\sigma > m^{\leftarrow}(x) | I_0 = k] \approx \Gamma(\beta_k + 1)/\Phi_k\left(m^{\leftarrow}(x)\right). \end{aligned}$$

This result also provides a numerical method to approximate the miss probabilities. It makes the computation feasible for complex settings, e.g., when the data sizes are correlated with the popularity distributions. In practice, once we have the information about the data sizes s_i and the corresponding data popularities p_i°, e.g., from the trace, we can always explicitly express $m(x)$, since i only takes a finite number of values in this case. Then, we can evaluate $m^{\leftarrow}(x)$ numerically; see Section 5. Explicit expressions for $m^{\leftarrow}(x)$ are derived for some cases in Section 3.1. Note that $m^{\leftarrow}(x)$ is tightly related to the characteristic time approximation [28, 38]; see Section 3.3.

3 MULTIPLE COMPETING FLOWS

In this section, we rigorously characterize the miss probability of a given request flow, say flow k, when it is mixed with other competing flows that share the same cache in Section 3.1. In Section 3.2, we provide a method to calculate $m(x)$ for multiple flows based on a decomposition property.

3.1 Asymptotic miss ratios

The miss probability of flow k, for a cache size x, is represented by a conditional probability $\mathbb{P}[C_0 > x | I_0 = k]$. Recall $\bar{s} = \sup_i s_i < \infty$ and that $p_i^{\circ} = \mathbb{P}[R_0 = d_i]$ is defined for the mixed flow. Note $m(z) \to \infty$ as $z \to \infty$. By the theory of regularly varying functions [20], a function $l(x) : \mathbb{R}^+ \to \mathbb{R}^+$ is slowly varying if for any $\lambda > 0, l(\lambda x)/l(x) \to 1$ as $x \to \infty$; and $\Phi(x) = x^{\beta} l(x)$ is called regularly varying of index β.

Assume that, for a function $0 < \delta(x) \le 1$ and $\epsilon > 0$,

$$\lim_{x \to \infty} \frac{m^{\leftarrow}\left((1 + \epsilon\delta(x))\, x\right)}{m^{\leftarrow}(x)} = f(\epsilon) \quad \text{with} \quad \lim_{\epsilon \to 0} f(\epsilon) = 1. \tag{13}$$

The function $\delta(x)$ characterizes how fast $m^{\leftarrow}(z)$ grows, and thus $\delta(x)$ should be selected to be as large as possible while still satisfying (13). For example, when $m^{\leftarrow}(x)$ is regularly varying, e.g., $m^{\leftarrow}(x) = x^{\beta}$, we can let $\delta(x) = 1$, implying $f(\epsilon) = (1 + \epsilon)^{\beta}, g(\epsilon) = 0$. When $m^{\leftarrow}(x) = e^{x^{\xi}}, 0 < \xi < 1$, we can pick $\delta(x) = x^{-\xi}$, implying $f(\epsilon) = \lim_{x \to \infty} e^{\left(x + \epsilon x^{1-\xi}\right)^{\xi}} / e^{x^{\xi}} = e^{\epsilon\xi}$. Both satisfy

$\lim_{\epsilon\to 0} f(\epsilon) = 1$. Note that in these examples $\delta(x)$ satisfies the following condition: there exist $h_2 > h_1 > 0, h_4 > h_3 > 0$ and x_0, for $x > x_0$,

$$h_1 < \frac{\delta(x)}{\delta(x+\epsilon\delta(x))} < h_2, h_3 < \frac{\delta(x-\epsilon\delta(x))}{\delta(x)} < h_4. \tag{14}$$

Theorem 3.1. *Consider M flows sharing a cache. Under assupmtions (5), (13) and (14), for $\Phi_k(x) \sim x^{\beta_k} l_k(x)$, $1 \le k \le M$ and $\overline{\lim}_{x\to\infty} \log\left(m^{\leftarrow}(x)\right)/(\delta^2(x)x) = 0$, we have, for $\beta_k \ge 0$ (when $\beta_k = 0$, $l_k(x)$ is eventually non-decreasing), as $x \to \infty$,*

$$\mathbb{P}[C_0 > x | I_0 = k] \sim \frac{\Gamma(\beta_k + 1)}{\Phi_k(m^{\leftarrow}(x))}. \tag{15}$$

Theorem 3.1 is the rigorous version of the result described in (7). The proof is presented in Section 7.2. This theorem assumes an independent reference model, but the result can be generalized to dependent requests [72]; see also [59, 70, 81]. Based on Theorem 3.1, we can easily derive some corollaries. We begin with the special case when there is only a single flow k in service and all data items are of the same size $s_i \equiv 1$. For a single flow k, we simplify the notation by $\mathbb{P}[R_0 > x | I_0 = k] = \mathbb{P}[R_0 > x]$ and $\mathbb{P}[C_0 > x | I_0 = k] = \mathbb{P}[C_0 > x]$. Theorem 3.1 recovers the results in [56, 59] for Zipf's law

$$p_i^{(k)} = q_i^{(k)} \sim c/i^{\alpha}, \alpha > 1. \tag{16}$$

Our result enhances (16) in three aspects. First, we study multiple flows ($p_i^{(k)} \neq q_i^{(k)}$) that can have overlapped data items. The requested data items can also have different sizes. Second, we address the case $\alpha = 1$ (then c needs to be replaced by $l(i)$ as in (17)), and the results in [56, 59] assume $\alpha > 1$. For $\alpha < 1$, we need to assume that only a finite number of data items are considered in the popularity distribution; otherwise the distribution does not exist. Due to this difference, the asymptotic result in (15) is accurate only for large x when $\alpha \ge 1$. However, the insights obtained in this paper can be extended to the case $0 < \alpha < 1$ using the analytical results developed in [72]; see also [19, 57]. Third, our result can derive the asymptotic miss probability for a large class of popularity distributions, e.g., Weibull with varying data item sizes. Corollary 3.2 extends the results of Theorem 3 in [56] under the condition (16) to regularly varying probabilities

$$p_i^{(k)} \sim l(i)/i^{\alpha}, \alpha \ge 1, \tag{17}$$

with $l(\cdot)$ being a slowly varying function, e.g., $l(x) = \log x$.

Corollary 3.2. *Consider a single flow with $s_i \equiv 1$ and $p_i^{(1)} \sim l(i)/i^{\alpha}$, $\alpha > 1$. Let $l_1(x) = l(x)^{-1/\alpha}$ and $l_{n+1} = l_1(x/l_n(x))$, $n \ge 1$. If $l_{n_0}(x) \sim l_{n_0+1}(x)$ as $x \to \infty$ for some n_0, then*

$$\lim_{x\to\infty} \frac{\mathbb{P}[C_0 > x]}{\mathbb{P}[R_0 > x]} = (1 - 1/\alpha)\left(\Gamma(1 - 1/\alpha)\right)^{\alpha}. \tag{18}$$

Proof. First we provide a proof for the special case $l(x) = c$, which was proved in Theorem 3 of [56]. The proof for a general $l(x)$ is presented in Section 7.3.

Note that $p_x^{(k)} \sim c/x^{\alpha}$ and

$$\mathbb{P}[R_0 > x] = \sum_{i \ge x} p_i^{(1)} \sim \int_x^{\infty} \frac{c}{t^{\alpha}} dt = \frac{c}{(\alpha - 1)x^{\alpha - 1}}. \tag{19}$$

From (4), we obtian $\Phi_1(x) \sim (\alpha-1)c^{-1/\alpha}x^{1-1/\alpha}$. In addition, since

$$\begin{aligned} m(z) &\sim \sum_{i\geq 1}\left(1-\exp\left(-\frac{cz}{i^\alpha}\right)\right) \sim \int_1^\infty \left(1-\exp\left(-\frac{cz}{t^\alpha}\right)\right)dt \\ &\sim \Gamma(1-1/\alpha)c^{1/\alpha}z^{1/\alpha}, \end{aligned}$$

we have the inverse $m^{\leftarrow}(z) \sim z^\alpha/\left(c\Gamma(1-1/\alpha)^\alpha\right)$. Picking $\delta(x)=1$, it is easy to verify (14) and $\overline{\lim}_{x\to\infty}\log m^{\leftarrow}(x)/\delta(x)^2 x = 0$. Therefore, by Theorem 3.1, we obtain, as $x\to\infty$,

$$\mathbb{P}[C_0 > x] \sim \frac{\Gamma(2-1/\alpha)\Gamma(1-1/\alpha)^{\alpha-1}}{\alpha-1}\frac{c}{x^{\alpha-1}}. \tag{20}$$

Combining (20) and (19) finishes the proof. □

Corollary 3.3. *For a single flow with requests following a heavy-tailed Weibull distribution* $p_i^{(k)} \sim c\exp\left(-i^\xi\right)$, $0<\xi<1/3$ *and* $s_i\equiv 1$, *we have, for a Euler's constant* $\gamma = 0.5772\cdots$,

$$\lim_{x\to\infty}\frac{\mathbb{P}[C_0>x]}{\mathbb{P}[R_0>x]} = e^\gamma. \tag{21}$$

Proof. Since ce^{-x^ξ} is a decreasing function in x, we have

$$\int_x^\infty ce^{-y^\xi}dy \leq \sum_{i=x}^\infty c\exp\left(-i^\xi\right) \leq \int_{x-1}^\infty ce^{-y^\xi}dy. \tag{22}$$

Changing the variable $z=y^\xi$ and using the property of incomplete gamma function, we obtain

$$\int_x^\infty ce^{-y^\xi}dy = \int_{x^\xi}^\infty \frac{c}{\xi}z^{1/\xi-1}e^{-z}dz \sim \frac{c}{\xi}x^{1-\xi}e^{-x^\xi}, \tag{23}$$

which implies, for $0<\xi<1$,

$$\Phi_k(x) \sim \xi\left(\log(cx)\right)^{1-1/\xi}x. \tag{24}$$

Using Lemma 6 in [56], we obtain

$$m^{\leftarrow}(z) \sim e^{-\gamma}e^{z^\xi}/c. \tag{25}$$

Picking $\delta(x)=x^{-\xi}>0$, for $0<\xi<1/3$, it is easy to verify $\overline{\lim}_{x\to\infty}\log m^{\leftarrow}(x)/x^{1-2\xi}=0$ and (14). Combining (24) and (25), by Theorem 3.1, we derive

$$\mathbb{P}[C_0>x] \sim \frac{e^\gamma c}{\xi}x^{1-\xi}e^{-x^\xi}, \tag{26}$$

which, using (22) and (23), proves (21). □

3.2 Decomposition property

For multiple request flows sharing a single cache space, instead of calculating p_i°, $i\geq 1$ and deriving $m(x)$ from (6), we prove a decomposition property that simplifies the computation of $m(x)$. Let $P=\left(p_i^\circ, i\geq 1\right)$ be constructed from a set of distributions $Q^{(k)}=\left(q_i^{(k)}, i\geq 1\right)$ according to probabilities ν_k, $\sum_k \nu_k = 1$. Specifically, a random data item following the distribution P is generated by sampling from the distribution $Q^{(k)}$ with a probability ν_k. Since two flows k_1, k_2 have no overlapped data items, we have $\mathbb{P}\left[R_0=d_i^{(k_1)}\middle| I_0=k_2\right]=0$. Therefore, according to (3), $\left(p_i^\circ, i\geq 1\right)$ can be represented by an unordered list,

$$\left(\left(\nu_k q_i^{(k)}, k+i=m\right), m=2,3,4,\cdots\right). \tag{27}$$

Let $\bar{m}(z) = \sum_{i=0}^{\infty} s_i \left(1 - \exp(-p_i^{\circ} z)\right)$. Lemma 3.4 shows a decomposition property for $m(z)$ and $\bar{m}(z) \sim m(z)$ under certain conditions. Let $m^{(k)}(z) = \sum_{i=0}^{\infty} s_i^{(k)} \left(1 - \left(1 - q_i^{(k)}\right)^z\right)$ and $\bar{m}^{(k)}(z) = \sum_{i=0}^{\infty} s_i^{(k)} \left(1 - \exp\left(-q_i^{(k)} z\right)\right)$. It is often easier to compute $\bar{m}^{(k)}(z)$ than $m^{(k)}(z)$.

Lemma 3.4. *Without overlapped data items, if, for either $g(x) = m^{(k)}(x)$ or $g(x) = \bar{m}^{(k)}(x)$, we have $\overline{\lim}_{x\to\infty} g((1+\delta)x)/g(x) = f^{(k)}(\delta)$, $0 < \delta < 1$ with $\lim_{\delta\to 0} f^{(k)}(\delta) = 1$, then, as $z \to \infty$,*

$$m^{(k)}(z) \sim \bar{m}^{(k)}(z), \tag{28}$$

and

$$\bar{m}(z) \sim \sum_k \bar{m}^{(k)}(\nu_k z) \sim \sum_j m^{(k)}(\nu_k z) \sim m(z). \tag{29}$$

The proof of Lemma 3.4 is presented in Section 7. It can be used to compute $m(x)$ for multiple flows sharing the same cache. Applying Lemma 3.4 and Theorem 3.1, we derive the miss probability for each flow in the following corollary.

Corollary 3.5. *Consider M flows without overlapped data, satisfying $\mathbb{P}[R_0 = d_x^{(k)} | I_0 = k] \sim c_k/x^{\alpha_k}$, $1 \le k \le M$ and $\mathbb{P}[I_0 = k] = \nu_k$, $\sum_{k=1}^{M} \nu_k = 1$. Assume that the data items of flow k have identical sizes, i.e. $s_i^{(k)} = s^{(k)}, i \ge 1$. For $\widetilde{\alpha}_1 \triangleq \min_{1\le k\le n} \alpha_k$ and $S_1 = \{k \in \mathbb{Z} | \alpha_k = \widetilde{\alpha}_1, 1 \le k \le M\}$, we have, for $k \in S_1$,*

$$\mathbb{P}[C_0 > x | I_0 = k] \sim \frac{\Gamma(2 - 1/\widetilde{\alpha}_1)}{\widetilde{\alpha}_1 - 1} \frac{{\gamma_1}^{\widetilde{\alpha}_1 - 1}}{(\nu_k c_k)^{1 - \frac{1}{\widetilde{\alpha}_1}}} \frac{c_k}{x^{\widetilde{\alpha}_1 - 1}},$$

and for $k \in {S_1}^c$,

$$\mathbb{P}[C_0 > x | I_0 = k] \sim \frac{\Gamma(2 - 1/\alpha_k)}{\alpha_k - 1} \frac{{\gamma_1}^{\widetilde{\alpha}_1 - \frac{\widetilde{\alpha}_1}{\alpha_k}}}{(\nu_k c_k)^{1 - \frac{1}{\alpha_k}}} \frac{c_k}{x^{\widetilde{\alpha}_1 - \frac{\widetilde{\alpha}_1}{\alpha_k}}},$$

where

$$\gamma_1 = \Gamma(1 - 1/\widetilde{\alpha}_1) \sum_{k \in S_1} s^{(k)} (c_k \nu_k)^{1/\widetilde{\alpha}_1}. \tag{30}$$

Proof. For flow k, $1 \le k \le n$, we have

$$\Phi_k(x) \sim (\alpha_k - 1) c_k^{-1/\alpha_k} (\nu_k x)^{1 - 1/\alpha_k},$$
$$m^{(k)}(z) \sim s^{(k)} \Gamma(1 - 1/\alpha_k) (c_k z)^{1/\alpha_k}.$$

Using the decomposition property of Lemma 3.4, $m(z)$ is asymptotically determined by flows with indices in S_1,

$$\begin{aligned} m(z) &\sim \sum_{k=1}^{M} m^{(k)}(z) \sim \sum_{k=1}^{M} s^{(k)} \Gamma(1 - 1/\alpha_k) (c_k \nu_k z)^{1/\alpha_k} \\ &\sim \sum_{k \in S_1} s^{(k)} (c_k \nu_k)^{1/\widetilde{\alpha}_1} \Gamma(1 - 1/\widetilde{\alpha}_1) z^{1/\widetilde{\alpha}_1} \sim \gamma_1 z^{1/\widetilde{\alpha}_1}, \end{aligned}$$

implying

$$m^{\leftarrow}(z) \sim z^{\widetilde{\alpha}_1} / \gamma_1^{\widetilde{\alpha}_1}. \tag{31}$$

Now, by Theorem 3.1, we can prove the corollary after straightforward computations. □

Corollary 3.5 approximates the miss probabilities for multiple flows with different α_k when the cache size $x \to \infty$. When the cache size is small, this approximation is not accurate. One primary reason is that the flows in ${S_1}^c$ are not negligible when computing $m(z)$ for a small z. In order to improve the accuracy, we consider a larger set of dominating flows. Let $\widetilde{\alpha}_2 \triangleq \min_{k \in {S_1}^c} \alpha_k$ denote the second smallest value among all α_k's when S_1^c is not empty. For $S_2 = \{k \in \mathbb{Z} | \alpha_k = \widetilde{\alpha}_2, 1 \le k \le n\}$, we consider all flows in the set $S_1 \cup S_2$, and derive

$$m(z) \sim \sum_{k \in S_1} s^{(k)} (c_k \nu_k)^{1/\widetilde{\alpha}_1} \Gamma(1 - 1/\widetilde{\alpha}_1) z^{1/\widetilde{\alpha}_1} + \sum_{k \in S_2} s^{(k)} (c_k \nu_k)^{1/\widetilde{\alpha}_2} \Gamma(1 - 1/\widetilde{\alpha}_2) z^{1/\widetilde{\alpha}_2}.$$

The inverse function of $m(z)$ can be better approximated by

$$m^{\leftarrow}(z) \sim z^{\widetilde{\alpha}_1} \Big/ \left(\gamma_1 + \gamma_2 (z/\gamma_1)^{\widetilde{\alpha}_1/\widetilde{\alpha}_2 - 1}\right)^{\widetilde{\alpha}_1}, \tag{32}$$

where $\gamma_2 = \Gamma(1 - 1/\widetilde{\alpha}_2) \sum_{k \in S_2} s^{(k)} (c_k \nu_k)^{1/\widetilde{\alpha}_2}$ and γ_1 is defined in (30). We obtain more accurate numerical results for miss probabilities using (32) instead of (31) especially when the cache size is small, though the expressions in (31) and (32) are asymptotically equivalent. Experiments 2 in Section 5 validates this approximation. Alternatively, we also resort to numerical methods (e.g., binary search) to directly evaluate $m^{\leftarrow}(z)$ from $m(z)$ for more complex cases.

3.3 Connection to the characteristic time approximation

The miss probability of LRU algorithm has been extensively studied using the characteristic time approximation [28, 38]. Now we show that the characteristic time approximation is asymptotically accurate under certain conditions. A similar result has been proved for a fluid limit and for Zipf's law with $\alpha > 1$ in [56] and a mathematical explanation has been provided in [46]; also see a validity argument in [82]. For multiple flows, the overall miss probability computed by the characteristic time approximation is

$$\mathbb{P}_{CT}[C_0 > x | I_0 = k] = \sum_{i=1}^{\infty} q_i^{(k)} e^{-p_i^{(k)} T}, \tag{33}$$

where T is the unique solution to $\sum_{i=1}^{\infty} s_i (1 - e^{-p_i^{\circ} T}) = x$. Notably, applying Lemma 3.4, we have $T \sim m^{\leftarrow}(x)$, as $x \to \infty$.

Theorem 3.6. *Under the conditions of Theorem 3.1, we have, as* $x \to \infty$,

$$\mathbb{P}_{CT}[C_0 > x | I_0 = k] \sim \mathbb{P}[C_0 > x | I_0 = k] \sim \frac{\Gamma(1 + \beta_k)}{\Phi_k(m^{\leftarrow}(x))}. \tag{34}$$

The proof of Theorem 3.6 is presented in Section 7.4.

Theorem 3.6 shows that the miss ratios in (33) and Theorem 3.1 are asymptotically equal. Note that the characteristic time approximation does not fully exploit tail properties of popularity distributions. By introducing a functional relationship $\Phi_k(\cdot)$, Theorem 3.1 explicitly characterizes the asymptotic miss ratios based on the popularity distributions of multiple competing flows. This explicit form reduces the computation cost incurred by the long summation in (33).

4 POOLING AND SEPARATION

We first characterize the self-organizing behavior of LRU caching for multiple flows in Section 4.1. Then, we study how the interactions of competing flows impact the individual ones in Section 4.2. The consequences of overlapped data items across different flows are investigated in Section 4.3. Based on the insights, we discuss engineering implications in Section 4.4.

A pooling scheme serves the M request flows jointly using the cache space of size x. A separation scheme divides the cache space x into M parts according to fractions $\{u_k\}_{1\le k\le M}$, $\sum_{k=1}^{M} u_k = 1$, and allocates $u_k x$ to flow k.

4.1 Self-organizing behavior of pooling

Based on the asymptotic miss ratios derived in Theorem 3.1, we show that, when multiple flows have similar distributions and identical data item sizes, resource pooling asymptotically gives the best overall hit ratio achieved by the optimal separation scheme. Otherwise, the optimal separation scheme results in a better overall miss ratio. Note that the optimal separation scheme is static while the pooling scheme is adaptive without any parameter tuning or optimization. This explains why pooling is better in Fig. 3. Let $\mathbb{P}_s^*[C_0 > x]$ and $\mathbb{P}_p[C_0 > x]$ denote the overall miss probabilities under the optimal separation $\{u_k^*\}$ and under resource pooling, respectively.

Theorem 4.1. *For M flows without overlapped data, following $\mathbb{P}\left[R_0 = d_x^{(k)} \middle| I_0 = k\right] \sim c_k/x^{\alpha_k}$, $\alpha_k > 1$, $1 \le k \le M$ and the data items of flow k having the same size $s_i^{(k)} = s^{(k)}$, we have*

$$\lim_{x\to\infty} \mathbb{P}_p[C_0 > x]/\mathbb{P}_s^*[C_0 > x] \ge 1, \tag{35}$$

and the equality holds if and only if $s^{(1)} = s^{(2)} = \cdots = s^{(M)}$.

This result explains the simulation in Fig.4 when data item sizes are different. In practice, data item sizes vary, and they can be considered approximately equal if within the same range, as used by slabs of Memcached [9, 69]. Note that Theorem 4.1 only characterizes an asymptotic result. When the cache size is not large enough and α_k's are different, resource pooling can be worse than the optimal separation, as studied in [29]. As commented after Corollary 3.5, a better approximation for small cache sizes is to use Theorem 3.1 by numerically evaluating $m^{\leftarrow}(x)$. Theorem 4.1 also shows that when data item sizes vary significantly, resource pooling could be worse than separation, as illustrated in Fig. 4.

Proof. First, we assume $\alpha_k = \alpha$, $1 \le k \le M$. To characterize resource separation, by Theorem 3.1, we obtain

$$\mathbb{P}_s[C_0 > x] = \sum_{k=1}^{M} \mathbb{P}[I_0 = k]\mathbb{P}_s[C_0 > u_k x | I_0 = k] \sim \sum_{k=1}^{M} \nu_k c_k \frac{\Gamma(1-1/\alpha)^\alpha}{\alpha} \left(\frac{s^{(k)}}{u_k x}\right)^{\alpha-1}. \tag{36}$$

Since the optimal separation method $u^* = (u_1^*, u_2^*, \cdots, u_k^*)$ minimizes the overall asymptotic miss probability, we have

$$\begin{aligned} \text{minimize} \quad & \sum_{k=1}^{M} \nu_k c_k \frac{\Gamma(1-1/\alpha)^\alpha}{\alpha} \left(\frac{s^{(k)}}{u_k x}\right)^{\alpha-1} \\ \text{subject to} \quad & \sum_{k=1}^{M} u_k = 1, u_k \ge 0. \end{aligned}$$

The solution of this convex optimization problem satisfies the KKT conditions, and therefore, for $\forall i, j$ with $1 \le i, j \le M$,

$$c_i \nu_i \left(s^{(i)}\right)^{\alpha-1} / u_i^{*\alpha} = c_j \nu_j \left(s^{(j)}\right)^{\alpha-1} / u_j^{*\alpha},$$

resulting in

$$u_k^* = \frac{(c_k \nu_k)^{1/\alpha} \left(s^{(k)}\right)^{1-1/\alpha}}{\sum_{i=1}^{M} (c_i \nu_i)^{1/\alpha} \left(s^{(i)}\right)^{1-1/\alpha}}, \qquad k = 1, 2, \cdots, M. \tag{37}$$

From (36) and (37), we obtain

$$\mathbb{P}_s^*[C_0 > x] \sim \frac{\Gamma(1-1/\alpha)^\alpha}{\alpha x^{\alpha-1}} \left(\sum_{k=1}^M (c_k \nu_k)^{1/\alpha} (s^{(k)})^{1-1/\alpha} \right)^\alpha. \tag{38}$$

To study resource pooling, we obtain, by Corollary 3.5,

$$\begin{aligned}
\mathbb{P}_p[C_0 > x] &= \sum_{k=1}^M \mathbb{P}[I_0 = k] \mathbb{P}_p[C_0 > x | I_0 = k] \\
&\sim \sum_{k=1}^M \nu_k \frac{\Gamma(1-1/\alpha)^\alpha}{\alpha} \frac{\left(\sum_{i=1}^M (c_i \nu_i)^{1/\alpha} s^{(i)} \right)^{\alpha-1}}{{\nu_k}^{1-1/\alpha}} \frac{{c_k}^{1/\alpha}}{x^{\alpha-1}} \\
&= \frac{\Gamma(1-1/\alpha)^\alpha}{\alpha x^{\alpha-1}} \left(\sum_{k=1}^M (c_k \nu_k)^{1/\alpha} \right) \left(\sum_{i=1}^M (c_i \nu_i)^{1/\alpha} s^{(i)} \right)^{\alpha-1},
\end{aligned}$$

which, using (38) and Hölder's inequality, proves (35). The equality holds if and only if $s^{(1)} = s^{(2)} = \cdots = s^{(n)}$.

Now, if α_k's are not identical, let $\widetilde{\alpha}_1 = \min_{1 \le k \le n} \alpha_k$ and $S_1 = \{k \in \mathbb{Z} | \alpha_k = \widetilde{\alpha}_1, 1 \le k \le n\}$. By Corollary 3.5, we have, for resource pooling,

$$\mathbb{P}_p[C_0 > x] = \sum_{k=1}^M \mathbb{P}[I_0 = k] \mathbb{P}_p[C_0 > x | I_0 = k] \sim \sum_{k \in S_1} \mathbb{P}[I_0 = k] \mathbb{P}_p[C_0 > x | I_0 = k].$$

For separation, by (18), we have, as $x \to \infty$,

$$\mathbb{P}_s[C_0 > u_k x | I_0 = k] \sim c_k \frac{\Gamma(1-1/\alpha_k)^{\alpha_k}}{\alpha_k} \left(\frac{s^{(k)}}{u_k x} \right)^{\alpha_k - 1}.$$

Thus, the overall miss probability is

$$\begin{aligned}
\mathbb{P}_s[C_0 > x] &= \sum_{k=1}^M \mathbb{P}[I_0 = k] \mathbb{P}_s[C_0 > u_k x | I_0 = k] \sim \sum_{k \in S)1} \mathbb{P}[I_0 = k] \mathbb{P}_s[C_0 > u_k x | I_0 = k] \\
&\sim \sum_{k \in S_1} \nu_k c_k \frac{\Gamma(1-1/\alpha_k)^{\alpha_k}}{\alpha_k} \left(\frac{s^{(k)}}{u_k x} \right)^{\alpha_k - 1}.
\end{aligned}$$

Thus, the same arguments for the case $\alpha_k = \alpha$ can be repeated to prove (35) in this case. □

4.2 Impacts on individual flows

When the QoS (quality-of-service) of individual flows is important, we need to guarantee the miss ratio of each flow. The following theorem shows that, for each flow, cache pooling asymptotically achieves the same miss ratio as the optimal separation under certain conditions. Interestingly, the miss ratios of multiple competing flows decrease according to $c_k^{1/\alpha} \nu_k^{1/\alpha - 1}, 1 \le k \le M$ when sharing the same cache.

Corollary 4.2. *For M flows under the conditions of Theorem 4.1 with $s^{(1)} = s^{(2)} = \cdots = s^{(M)}$ and $\alpha_k = \alpha$, we have*

$$\lim_{x \to \infty} \mathbb{P}_p[C_0 > x | I_0 = k] / \mathbb{P}_s^*[C_0 > u_k^* x | I_0 = k] = 1.$$

Furthermore, the miss ratios of any two flows i, j satisfy

$$\lim_{x\to\infty}\frac{\mathbb{P}_p[C_0 > x|I_0 = i]}{\mathbb{P}_p[C_0 > x|I_0 = j]} = \lim_{x\to\infty}\frac{\mathbb{P}^*_s[C_0 > u_i^* x|I_0 = i]}{\mathbb{P}^*_s[C_0 > u_j^* x|I_0 = j]} = \frac{c_i^{1/\alpha}\nu_i^{1/\alpha-1}}{c_j^{1/\alpha}\nu_j^{1/\alpha-1}}.$$

The proof of this corollary is based on the same arguments in the proof of Theorem 4.1. This result quantifies the empirical observation [9] that mixing multiple flows benefits the ones with large arrival rates at the expense of the others with small arrival rates. It also shows that the popularity distributions need to be considered if $c_i \neq c_j$. Therefore, if arrival rates differ significantly, mixing flows requires extra caution. Simulations for validating Corollary 4.2 is in Section 5.

4.3 Overlapped data items

In this section, we show that when overlapped data items exceed certain levels, resource pooling can even improve the performance of every flow. Since overlapped data items across more than two flows are complicated, we only consider two flows with unit-sized data items. Notably, there always exists a good region of parameters such that the miss probabilities of both flows under pooling are better than under separation; see Experiment 3 in Section 5.

Since the requested data items can overlap (see Fig. 5), we introduce 3 disjoint classes of data items, A, B and D for the two flows. Flow 1 and 2 request data items from class $A = \{d_i^{(A)}, i = 1, 2, \cdots\}$ and class $B = \{d_i^{(B)}, i = 1, 2, \cdots\}$, respectively. Class $D = \{d_i^{(D)}, i = 1, 2, \cdots\}$ represents the common data items that are requested by both flow 1 and 2. We use $J_n = A, J_n = B, J_n = D$ to indicate that the request n is for class A, B and D, respectively. Let $\mathbb{P}[J_n = A \mid I_n = 1] = p_A^{(1)}, \mathbb{P}[J_n = D \mid I_n = 1] = p_D^{(1)}$ with $p_A^{(1)} + p_D^{(1)} = 1$, and $\mathbb{P}[J_n = B \mid I_n = 2] = p_B^{(2)}, \mathbb{P}[J_n = D \mid I_n = 2] = p_D^{(2)}$ with $p_B^{(2)} + p_D^{(2)} = 1$. Class A and B have $\mathbb{P}\left[R_0 = d_x^{(A)}|I_0 = 1, J_0 = A\right] \sim c_A/x^\alpha, \mathbb{P}[R_0 = d_x^{(B)}|I_0 = 2, J_0 = B] \sim c_B/x^\alpha$. For class D, we assume that $\mathbb{P}\left[R_0 = d_x^{(D)}|I_0 = k, J_0 = D\right] \sim c_D/x^\alpha, k = 1, 2$.

An optimal separation scheme has been proposed in [34] to serve classes A, B, D in three isolated parts of the whole cache space. Since the three classes do not have overlapped data items, Theorem 4.1 implies that the optimal separation is asymptotically equivalent to pooling. However, this isolation scheme requires a lot of tracking information, and is difficult to implement in practice. We consider a practical constraint that a flow is the smallest unit that cannot be further divided into sub-flows. In this case, the optimal separation is not always the best. In fact, it can be worse than resource pooling if enough data overlap is present.

For a static separation $u = (u_1, u_2)$, define a good region $\mathcal{G}_u$ for positive parameters $P = \left(\nu_1, \nu_2, c_A, c_B, p_A^{(1)}, p_B^{(2)}, p_D^{(1)}, p_D^{(2)}\right)$, which satisfy, for $p_D^* = p_D^{(1)}\nu_1 + p_D^{(2)}\nu_2 > 0$,

$$\frac{\left(\left(c_A p_A^{(1)}\right)^{1/\alpha} + \left(c_D p_D^{(1)}\right)^{1/\alpha}\right)^\alpha}{\left(\left(c_A p_A^{(1)}\nu_1\right)^{1/\alpha} + \left(c_B p_B^{(2)}\nu_2\right)^{1/\alpha} + \left(c_D p_D^*\right)^{1/\alpha}\right)^{\alpha-1}} > {u_1}^{\alpha-1}\left(\frac{\left(c_A p_A^{(1)}\right)^{1/\alpha}}{{\nu_1}^{1-1/\alpha}} + \frac{{c_D}^{1/\alpha} p_D^{(1)}}{{p_D^*}^{1-1/\alpha}}\right), \tag{39}$$

with another symmetric constraint that replaces $u_1, c_A, c_B, p_A^{(1)}, p_B^{(2)}, p_D^{(1)}$ in (39) with $u_2, c_B, c_A, p_B^{(2)}, p_A^{(1)}, p_D^{(2)}$, respectively. The following corollary shows that when the parameters satisfy $P \in \mathcal{G}_u$, both flows have smaller miss ratios by resource pooling than by the static separation u. Remarkably, the parameters in P for the optimal static separation u^* that minimizes the overall miss ratio (defined in Section 4.1) are always in the good region $\mathcal{G}_{u^*}$, although this region is defined to study the miss ratios of individual flows.

Corollary 4.3. *For any positive $u = (u_1, u_2)$, if $P \in \mathcal{G}_u$ and is strictly positive, then we have, for $k = 1, 2$,*

$$\lim_{x\to\infty} \mathbb{P}_p[C_0 > x | I_0 = k] / \mathbb{P}_s[C_0 > u_k x | I_0 = k] < 1.$$

Furthermore, if $u = u^*$, then we always have $P \in \mathcal{G}_{u^*}$.

The proof is a straightforward computation based on Theorem 3.1. This corollary also implies that $\mathcal{G}_u$ is always nonempty. We use simulations to validate Corollary 4.3 in Section 5.

4.4 Engineering Implications

Whether resource pooling or separation should be used for LRU caching is complicated and has no straight yes or no answers, depending on four critical factors: the popularity distributions, request rates, data item sizes and overlapped data across different flows. This problem becomes even more complicated due to engineering issues. However, there are still guidelines to improve the hit ratios.

Our analysis shows that for large cache spaces it is beneficial to jointly serve multiple flows if their data item sizes and popularity distributions are similar and their arrival rates do not differ significantly. Although the optimal static resource separation scheme has been shown to always theoretically achieve the best performance under certain assumptions [34], in practice the number of separate clusters deployed in service, e.g., Memcached clusters, is relatively small [9]. This may be partially attributed to the self-organizing behavior of LRU for resource pooling. As shown in Theorem 4.1, resource pooling can adaptively achieve the optimal resource allocation for multiple competing flows asymptotically when the data item sizes are equal. In practice two data items can be considered to have an approximately equal size if within the same range. This property is especially beneficial when the request statistics, including the distributions and rates, are time-varying. Nevertheless, we also point that, due to the impact of competing flows, careful separation could be necessary if we want to guarantee the miss ratios of individual flows for certain QoS requirements.

The current practice uses applications and domains to separate flows of requests into different cache spaces [9, 69]. One possible explanation is that the data item sizes, e.g., text and image objects, and request rates are quite different. This also suggests that there is still room for possible improvement. A more careful strategy based on the quantitative characterization may lead to optimal or near-optimal performance. For example, our analysis shows that distributions are important in determining the miss ratios. Thus, it appears to be beneficial if the statistics of different flows can be further exploited in practice.

5 EXPERIMENTS

We implement an LRU simulator using C++ and conduct extensive simulation experiments. First, we verify Theorem 3.1 for data items of varying sizes with distributions beyond Zipf's distributions. Next, we study the interactions among multiple competing flows on the same cache space. Last, we investigate the impact of overlapped data items across flows by verifying Corollaries 4.2 and 4.3. The numerical results based on analyses match accurately with the simulation experiments, even for small cache sizes and finitely many data items.

Experiment 1. Consider 3 flows of data item requests that share a single cache space. The varying data item sizes take different values that are also correlated with the popularity distributions. Set $q_1^{(k)} = 0.1$, and $q_i^{(k)} = c_k \log i / i^{\alpha_k}$ for $2 \le i \le N = 10^5$, $1 \le k \le 3$, which are beyond Zipf's distributions. Set $[\alpha_1, \alpha_2, \alpha_3] = [2.0, 2.2, 2.4]$, $[\nu_1, \nu_2, \nu_3] = [0.2, 0.3, 0.5]$. Then, $c_1 = (1 - q_1^{(1)})/(\sum_{i=2}^{N} (\log i) i^{-\alpha_1}) = 0.9601$, $c_2 = (1 - q_1^{(2)})/(\sum_{i=2}^{N} (\log i) i^{-\alpha_2}) = 1.4193$, $c_3 =$

$(1-q_1^{(3)})/(\sum_{i=2}^{N}(\log i)i^{-\alpha_3}) = 1.9910$. Let $d_i^{(k)}$ denote the i'th data item of flow k with size $s_i^{(k)}$, $1 \le k \le 3, 1 \le i \le 10^5$. Based on the empirical distribution [9], the data sizes for each flow are independently drawn from a generalized Pareto distribution with parameters $\xi = 0.1, \mu = 1, \sigma = 50$. To be correlated with the popularity distribution $q_i^{(k)}$, these data sizes are sorted in a way that $s_i^{(k)}$, $1 \le k \le 3$ are non-decreasing with respect to i. Note that $q_i^{(k)}$, $1 \le k \le 3$ are decreasing with respect to i. Theoretical miss probabilities are approximated by Theorem 3.1, where $m^{\leftarrow}(x)$ is evaluated by a binary search based on (6). The empirical miss probabilities and their theoretical approximations are plotted in Fig. 6. The good match validates Theorem 3.1 even when the popularities are beyond Zipf's distributions ($\approx \log i/i^{\alpha_k}$) and also correlated with the data sizes.

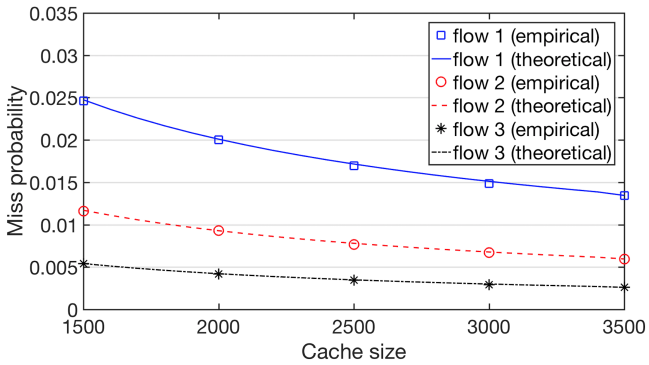


Fig. 6. Flows beyond Zipf's distributions with varying data sizes

Experiment 2. This experiment compares the miss ratios when a flow is served exclusively in a dedicated cache and when it shares the same cache with other flows. We show how one flow is impacted by other competing flows, through validating Corollary 3.5. Consider 10 flows without overlapped data items. Let $v_k = 0.1$ and $d_i^{(k)}, i = 1, 2, 3, \cdots$ be the data items of flow k for $1 \le k \le 10$. Data popularities of each flow are assumed to follow a Zipf's law, i.e. $\mathbb{P}[R_0 = d_i^{(k)} | I_0 = k] \sim c_k/i^{\alpha_k}, 1 \le k \le 10$. Let N_k be the number of data items of flow k. Set $\alpha_i = 2.5, 1 \le i \le 5$, $\alpha_j = 1.5, 6 \le j \le 10$, and $N_k = 10^6, 1 \le k \le 10$, and therefore, $c_i = (\sum_{x=1}^{N_1} x^{-\alpha_1})^{-1} = 0.7454$, $c_j = (\sum_{x=1}^{N_6} x^{-\alpha_6})^{-1} = 0.3831$. Note that we use the enhanced approximation (32), instead of (31), to compute $m^{\leftarrow}(x)$ when the cache size x is relatively small. The theoretical and empirical results for

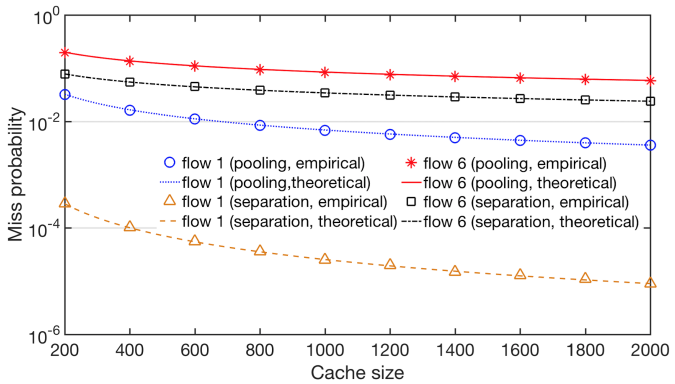
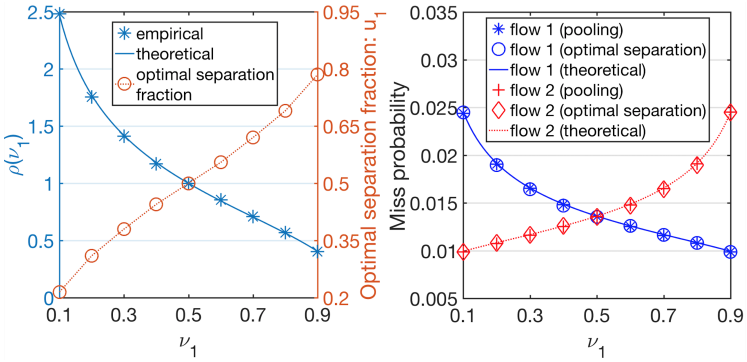


Fig. 7. Impacts among 10 flows

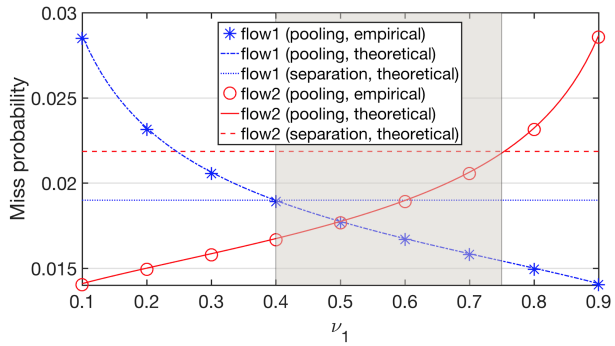
the miss probabilities are plotted in Fig. 7 when changing the cache capacity from 200 to 2000. Since

flows $1-5$ (respectively flows $6-10$) have the same popularity distribution and the same miss ratio, we only plot flow 1 (respectively flow 6). It can be observed that the empirical results match with the numerical results even when the cache size is relatively small. In this case, flow 1 has a miss probability tail that decays on the order of $1/x^{0.9}$, as shown by the curve with a label flow 1 (pooling, empirical). However, if flow 1 is served without others, as shown by the curve with a label flow 1 (separation, empirical), its probability tail only decays on the order of $1/x^{1.5}$. Therefore, in this case it is much worse for flow 1 (respectively flows $2-5$) to share with others than to be served exclusively. On the other hand, flow 6 (respectively flows $7-10$) is not significantly impacted when served together with other flows, since $\alpha_6 < \alpha_i, 1 \le i \le 5$.

Experiment 3. To address overlapped common data items, we simulate 2 flows with 3 classes of data A, B, D defined in Section 4.3, and use the same notation introduced therein. Let N_A, N_B, N_D be the numbers of data items of class A, B and D, respectively. Set cache size $x = 1000$, $N_A = N_B = N_D = 10^6$, $\alpha_A = \alpha_B = \alpha_D = 1.7$, $c_A = c_B = c_D = \left(\sum_{i=1}^{N_A} i^{-\alpha_A}\right)^{-1} = 0.4868$. First, we



(a) Without overlapped data (Corollary 4.2)



(b) With overlapped data (Corollary 4.3)

Fig. 8. Two flows sharing a server

assume these two flows have no overlapped data (i.e., $p_D^{(1)} = p_D^{(2)} = 0$). In Fig. 8(a), we plot $\rho(\nu_1) \triangleq \mathbb{P}_s^*[C_0 > u_1 x | I_0 = 1]/\mathbb{P}_s^*[C_0 > u_2 x | I_0 = 2]$ and the miss ratios for both flows under resource pooling and the optimal separation to validate Corollary 4.2. The simulations match with the theoretical results. Then, we assume these two flows have 20% overlapped data items (i.e., $p_D^{(1)} = p_D^{(2)} = 0.2$). In Fig. 8(b), we plot the miss ratios under resource pooling and under a static separation $(u_1, u_2) = (0.55, 0.45)$. It can be observed that in the shaded area $\nu_1 \in (0.40, 0.75)$, which is exactly the good region calculated by Corollary 4.3, both flows have lower miss ratios under resource pooling than under the static separation. This result validates Corollary 4.3. In presence

of overlapped data, there exists a good region where both flows have better hit ratios by pooling. However, when the arrival rates of these two flows are very different, i.e., $\nu_1 < 0.4$ or $\nu_1 > 0.75$, the flow with a lower arrival rate will be negatively impacted.

6 CONCLUSION

When designing a caching system shared by multiple request flows, should we use resource pooling or separation for better hit ratios? This paper develops a theoretical framework to answer this fundamental question. Roughly speaking, for flows with similar request distributions and data item sizes, with close arrival rates, and/or with enough overlapped data items, it is beneficial to jointly serve them by combining their allocated cache spaces together. However, for flows with disparate request distributions, i.e., probability tails decaying at different rates, or with clearly different arrival rates, isolating the cache spaces provides a guarantee for the hit ratios of individual flows. Otherwise, some of the flows could be negatively impacted, even severely penalized. Our results provide useful insights that can be exploited to potentially further improve the hit ratios of caching systems.

7 PROOFS

This section contains the details of the proofs.

7.1 Proof of Lemma 3.4

Without loss of generality, we assume that $\left(q_i^{(k)}\right)$ is a non-increasing sequence in i for each fixed k. We begin with $g(x) = \bar{m}(x)$. First, using the inequality $\left(1 - q_i^{(k)}\right)^z \le \exp(-q_i^{(k)} z)$, we obtain

$$m^{(k)}(z) \ge \sum_{i=0}^{\infty} s_i^{(k)} \left(1 - \exp\left(-q_i^{(k)} z\right)\right) = \bar{m}^{(k)}(z). \tag{40}$$

Next, for any $0 < \delta < 1$, there exists $x_\delta > 0$ such that $1 - x \ge e^{-(1+\delta)x}, 0 \le x \le x_\delta$. Thus, selecting i_δ with $q_{i_\delta}^{(k)} < x_\delta$, we have

$$\begin{aligned} m^{(k)}(z) &\le \left(\sum_{i=0}^{i_\delta} + \sum_{i>i_\delta}\right) s_i^{(k)} \left(1 - \left(1 - q_i^{(k)}\right)^z\right) \le i_\delta \bar{s} + \sum_{i>i_\delta} s_i^{(k)} \left(1 - \exp\left(-(1+\delta) q_i^{(k)} z\right)\right) \\ &\le i_\delta \bar{s} + \bar{m}^{(k)}((1+\delta)z), \end{aligned} \tag{41}$$

where the second last inequality uses $s_i^{(k)} \le \bar{s}$. Using (40), (41) and $\overline{\lim}_{x\to\infty} m^{(k)}((1+\delta)x)/m^{(k)}(x) \to 1$ as $\delta \to 0$, we prove (28).

Based on the representation of $\left(p_i^\circ, i \ge 1\right)$ in (27), we have

$$m(z) = \sum_{k=1}^{\infty} \sum_{i=1}^{\infty} s_i^{(k)} \left(1 - \left(1 - \nu_k q_i^{(k)}\right)^z\right). \tag{42}$$

Using the same arguments as in (40) and (41), we can prove

$$\bar{m}^{(k)}(\nu_k z) \sim \sum_{i=1}^{\infty} s_i^{(k)} \left(1 - \left(1 - \nu_k q_i^{(k)}\right)^z\right). \tag{43}$$

Using (42), (43) and applying (28), we finish the proof of (29) when $g(x) = \bar{m}(x)$. Slightly modifying the preceding arguments can prove (29) when $g(x) = m(x)$.

7.2 Proof of Theorem 3.1

In order to prove Theorem 3.1, we need to establish a lemma.

LEMMA 7.1. *For $\epsilon(x) = \epsilon\delta(x)$ as in (13) and $\bar{s} = \sup_i s_i < \infty$, we obtain*

$$\mathbb{P}\left[M(m^{\leftarrow}(x)) \geq (1+\epsilon(x))x\right] \leq e^{-(\epsilon(x))^2 x/4\bar{s}}. \tag{44}$$

PROOF. Define a Bernoulli random variable X_i, and let $X_i = 1$ to indicate that item d_i has been requested in $R_{-1}, R_{-2}, \cdots, R_{-n}$ and $X_i = 0$ otherwise. By Markov's inequality, for $\theta > 0$, we obtain, using $\mathbb{P}[X_i = 1] = p_i(n)$, $\mathbb{E}[e^{\theta s_i X_i}] = p_i(n)e^{\theta s_i} + 1 - p_i(n) = p_i(n)\left(e^{\theta s_i} - 1\right) + 1 \leq e^{p_i(n)\left(e^{\theta s_i}-1\right)}$ and independence of X_i's,

$$\begin{aligned}\mathbb{P}\left[M(n) \geq (1+\epsilon(m(n)))m(n)\right] &\leq \mathbb{E}\left[e^{\theta\sum_{i=1}^{\infty} s_i X_i}\right] / e^{(1+\epsilon(m(n)))\theta m(n)} \\ &\leq \exp\left(\sum_{i=1}^{\infty} p_i(n)\left(e^{\theta s_i}-1\right) - \theta(1+\epsilon(m(n)))\sum_{i=1}^{\infty} p_i(n)s_i\right).\end{aligned}$$

Using $e^x - 1 \leq (1+\xi)x, 0 < x < 2\xi/e^{\xi}, \xi > 0$, for $\theta = \epsilon(m(n))/(2\bar{s})$, we obtain, $e^{\theta s_i} - 1 \leq (1+\epsilon(m(n))/2)\theta s_i$. Therefore,

$$\mathbb{P}\left[M(n) \geq (1+\epsilon(m(n)))m(n)\right] \leq \exp\left(-\sum_{i=1}^{\infty}\frac{(\epsilon(m(n)))^2}{4\bar{s}}p_i(n)s_i\right),$$

which, by $\sum_{i=1}^{\infty} p_i(n)s_i = m(n)$, yields

$$\mathbb{P}\left[M(n) \geq (1+\epsilon(m(n)))m(n)\right] \leq e^{-(\epsilon(m(n)))^2 m(n)/4\bar{s}},$$

implying (44) by replacing $x = m(n)$. Using the same approach, we can prove

$$\mathbb{P}\left[M(m^{\leftarrow}(x)) \leq (1-\epsilon(x))x\right] \leq e^{-(\epsilon(x))^2 x/4\bar{s}}. \tag{45}$$

□

Next we prove Theorem 3.1.

PROOF. In the intuitive proof of Section 2, we have derived

$$\mathbb{P}[C_0 > x | I_0 = k] = \mathbb{P}[\sigma > M^{\leftarrow}(x) | I_0 = k]. \tag{46}$$

The whole proof consists of two steps. The first step is to show

$$\mathbb{P}[\sigma > n | I_0 = k] \sim \Gamma(\beta_k + 1)/\Phi_k(n), \tag{47}$$

for $\beta_k > 0$ and $\beta_k = 0$, respectively. The second step is to relate $M^{\leftarrow}(x)$ to $m^{\leftarrow}(x)$ as $x \to \infty$.
Step 1. First, we consider $\beta_k > 0$. Assume that $\Phi_k(x)$ is eventually absolutely continuous and strictly monotone, since, by Proposition 1.5.8 of [20], we can construct such a function

$$\Phi_k^*(x) = \beta_k \int_{x_0}^{x} \Phi_k(s)s^{-1}ds, \; x \geq x_0, \tag{48}$$

which, for x_0 large enough, satisfies, as $y \to \infty$,

$$\left(\sum_{i=y}^{\infty} q_i^{(k)}\right)^{-1} \sim \Phi_k\left(\left(p_y^{(k)}\right)^{-1}\right) \sim \Phi_k^*\left(\left(p_y^{(k)}\right)^{-1}\right).$$

Therefore, there exists x_0 such that for all $x > x_0$, $\Phi_k(x)$ has an inverse function $\Phi_k^{\leftarrow}(x)$. The condition (4) implies that, for $0 < \epsilon_1 < 1$, there exists i_{ϵ_1}, such that for $i > i_{\epsilon_1}$,

$$(1-\epsilon_1)\left(\sum_{j=i}^{\infty} q_j^{(k)}\right)^{-1} \leq \Phi_k\left(\left(p_i^{(k)}\right)^{-1}\right) \leq (1+\epsilon_1)\left(\sum_{j=i}^{\infty} q_j^{(k)}\right)^{-1}, \tag{49}$$

and thus, by choosing i_{ϵ_1} such that $1/p_{i_{\epsilon_1}}^{(k)} > x_0$, we obtain

$$\Phi_k^{\leftarrow}\left((1-\epsilon_1)\left(\sum_{j=i}^{\infty} q_j^{(k)}\right)^{-1}\right) \leq \left(p_i^{(k)}\right)^{-1} \leq \Phi_k^{\leftarrow}\left((1+\epsilon_1)\left(\sum_{j=i}^{\infty} q_j^{(k)}\right)^{-1}\right). \tag{50}$$

First, we will prove an *upper bound* for (47). Combining (8) and (50) yields, using $(1-p)^n \leq e^{-np}$,

$$\begin{aligned}\mathbb{P}\left[\sigma > n | I_0 = k\right] &= \left(\sum_{i=1}^{i_{\epsilon_1}} + \sum_{i=i_{\epsilon_1}+1}^{\infty}\right) q_i^{(k)}\left(1-p_i^{(k)}\right)^n \\ &\leq \left(1-p_{i_{\epsilon_1}}^{(k)}\right)^n + \sum_{i=i_{\epsilon_1}+1}^{\infty} q_i^{(k)} e^{-np_i^{(k)}} \triangleq I_1 + I_2. \end{aligned} \tag{51}$$

For $0 < \epsilon_2 \leq p_{i_{\epsilon_1}}^{(k)}$, integer n large enough, and any nonnegative integer $l \leq \lfloor \log n\epsilon_2 \rfloor$, we can find i_l such that $p_{i_l+1}^{(k)} \leq e^l/n \leq p_{i_l}^{(k)} \leq \epsilon_2$. Choose an integer m with $0 < m < \lfloor \log n\epsilon_2 \rfloor$. We have $i_0 > i_m > i_{\lfloor \log n\epsilon_2 \rfloor} > i_{\epsilon_1}$, and

$$\begin{aligned} I_2 &= \left(\sum_{i=i_{\epsilon_1}+1}^{i_{\lfloor \log n\epsilon_2 \rfloor}-1} + \sum_{i=i_{\lfloor \log n\epsilon_2 \rfloor}}^{i_m} + \sum_{i=i_m+1}^{\infty}\right) q_i^{(k)} e^{-np_i^{(k)}} \\ &\leq e^{-n\epsilon_2} + \sum_{l=m}^{\infty} e^{-e^l} \sum_{j=i_{l+1}+1}^{i_l} q_j^{(k)} + \sum_{j=i_m+1}^{\infty} q_j^{(k)} e^{-np_j^{(k)}} \\ &\triangleq I_{21} + I_{22} + I_{23}. \end{aligned} \tag{52}$$

We have $I_{23} = \sum_{j=i_m+1}^{\infty}\left(Q_j - Q_{j+1}\right) e^{-n/\Phi_k^{\leftarrow}\left((1+\epsilon_1)Q_j^{-1}\right)}$ for $Q_j = \sum_{i=j}^{\infty} q_i^{(k)}$. Since $e^{-n/\Phi_k^{\leftarrow}\left((1+\epsilon_1)u^{-1}\right)} \geq e^{-n/\Phi_k^{\leftarrow}\left((1+\epsilon_1)Q_j^{-1}\right)}$ for $\forall u \in (Q_{j+1}, Q_j)$, we have

$$I_{23} \leq \int_0^{Q_{i_m}} e^{-n/\Phi_k^{\leftarrow}\left((1+\epsilon_1)u^{-1}\right)} du \leq \int_0^{\epsilon_1} d\left(\frac{1+\epsilon_1}{\Phi_k(n/z)}\right) + \int_{\epsilon_1}^{e^m} e^{-z} d\left(\frac{1+\epsilon_1}{\Phi_k(n/z)}\right).$$

By Theorem 1.2.1 of [20] and (48), we obtain,

$$I_{23}\Phi_k(n) \lesssim (1+\epsilon_1)\epsilon_1^{\beta_k} + \int_{\epsilon_1}^{e^m} (1+\epsilon_1)\beta_k e^{-z} z^{\beta_k - 1} dz. \tag{53}$$

For I_{22}, using the same approach, we obtain

$$I_{22}\Phi_k(n) \lesssim \sum_{k=m}^{\infty} (1+\epsilon_1) e^{-e^k}\left(e^{k+1}\right)^{\beta_k} < \infty. \tag{54}$$

Combining (53) and (54), and then passing $\epsilon_1 \to 0$ and $m \to \infty$, we obtain, using $I_1 = o(1/\Phi_k(n))$ in (51),

$$\mathbb{P}[\sigma > n | I_0 = k]\Phi_k(n) \lesssim \int_0^{\infty} \beta_k e^{-z} z^{\beta_k - 1} dz = \Gamma(\beta_k + 1). \tag{55}$$

Now, we prove the *lower bound* for (47). By condition (5) we can choose i_{ϵ_1} large enough such that, for all $i > i_{\epsilon_1}$,

$$q_i^{(k)} \geq (1-\epsilon_1) q_{i-1}^{(k)}. \tag{56}$$

Using (8), (50) and the monotonicity of $\Phi_k^{\leftarrow}(\cdot)$, we obtain

$$\begin{aligned}
\mathbb{P}\left[\sigma > n | I_0 = k\right] &\geq \sum_{i=i_{\epsilon_1}+1}^{\infty} q_i^{(k)} \left(1-p_i^{(k)}\right)^n \\
&\geq (1-\epsilon_1) \sum_{i=i_{\epsilon_1}+1}^{\infty} \Delta Q_i \left(1 - \frac{1}{\Phi_k^{\leftarrow}\left((1-\epsilon_1)\left(Q_i\right)^{-1}\right)}\right)^n \\
&\geq (1-\epsilon_1) \int_0^{Q_{i_{\epsilon_1}}} \left(1 - 1/\Phi_k^{\leftarrow}\left((1-\epsilon_1)u^{-1}\right)\right)^n du.
\end{aligned} \tag{57}$$

where $\Delta Q_i = Q_{i-1} - Q_i = q_{i-1}^{(k)}$. For $W > 0$, choosing $i_n > i_\epsilon$ with $\Phi_k^{\leftarrow}\left((1-\epsilon_1)Q_{i_n}\right) = n/W$ and letting $z = n/\Phi_k^{\leftarrow}\left((1-\epsilon_1)u^{-1}\right)$, we obtain,

$$\begin{aligned}
\mathbb{P}[\sigma > n | I_0 = k]\Phi_k(n) &\geq (1-\epsilon_1)\Phi_k(n) \int_0^{Q_{i_n}} \left(1 - \frac{1}{\Phi_k^{\leftarrow}\left((1-\epsilon_1)/u\right)}\right)^n du \\
&\geq (1-\epsilon_1) \int_{\epsilon_1}^{W} \left(1-\frac{z}{n}\right)^n d\left(\frac{1-\epsilon_1}{\Phi_k(n/z)}\right).
\end{aligned} \tag{58}$$

From (58), by using the same approach as in deriving (53), we obtain, as $n \to \infty$,

$$\mathbb{P}[\sigma > n | I_0 = k]\Phi_k(n) \gtrsim (1-\epsilon_1) \int_{\epsilon}^{W} (1-\epsilon_1)\beta_k e^{-z} z^{\beta_k - 1} dz,$$

which, passing $W \to \infty$ and $\epsilon_1 \to 0$, yields

$$\mathbb{P}[\sigma > n | I_0 = k]\Phi_k(n) \gtrsim \int_0^{\infty} \beta_k e^{-z} z^{\beta_k - 1} dz = \Gamma(\beta_k + 1). \tag{59}$$

Combining (55) and (59) completes the proof of (47).

Up to now, we have proved (47) for $\beta_k > 0$. Using a similar approach, we can prove (47) for $\beta_k = 0$. For $\beta_k = 0$, we need to prove

$$\mathbb{P}\left[\sigma > n | I_0 = k\right] \sim 1/\Phi_k(n). \tag{60}$$

Recall that there exists x_0 such that $\Phi_k(x)$ is strictly increasing for $x > x_0$. For any positive integer n, we can find $\epsilon_3 \in (0,1)$ with $n/\epsilon_3 > x_0$. Because of the monotonicity of $p_i^{(k)}$, there exists an index i_{ϵ_3} such that $p_{i_{\epsilon_3}}^{(k)} \geq \epsilon_3/n$ and $p_i^{(k)} < \epsilon_3/n$ for all $i > i_{\epsilon_3}$. By choosing ϵ_3 sufficiently small such that $i_{\epsilon_3} > i_{\epsilon_1}$, we can derive the lower bound for the miss probability

$$\mathbb{P}\left[\sigma > n | I_0 = k\right] = \sum_{i=1}^{\infty} q_i^{(k)} \left(1-p_i^{(k)}\right)^n \geq \left(1-\frac{\epsilon_3}{n}\right)^n \sum_{i=i_{\epsilon_3}+1}^{\infty} q_i^{(k)}.$$

Using (49) and (56), we obtain

$$\mathbb{P}\left[\sigma > n | I_0 = k\right] \geq \left(1 - \frac{\epsilon_3}{n}\right)^n (1-\epsilon_1) \sum_{i=i_{\epsilon_3}}^{\infty} q_i^{(k)} \geq \left(1 - \frac{\epsilon_3}{n}\right)^n \frac{(1-\epsilon_1)^2}{\Phi_k\left(\left(p_{i_{\epsilon_3}}^{(k)}\right)^{-1}\right)}$$
$$\geq \left(1 - \frac{\epsilon_3}{n}\right)^n \frac{(1-\epsilon_1)^2}{\Phi_k\left(\frac{n}{\epsilon_3}\right)},$$

implying

$$\mathbb{P}\left[\sigma > n | I_0 = k\right] \Phi_k(n) \geq \left(1 - \frac{\epsilon_3}{n}\right)^n (1-\epsilon_1)^2 \frac{\Phi_k(n)}{\Phi_k\left(\frac{n}{\epsilon_3}\right)}.$$

By passing $n \to \infty$ and $\epsilon_1, \epsilon_3 \to 0$, we obtain

$$\underline{\lim}_{n\to\infty} \mathbb{P}\left[\sigma > n | I_0 = k\right] \Phi_k(n) \geq 1. \tag{61}$$

Next, we prove the upper bound. The miss ratio can be bounded as

$$\mathbb{P}\left[\sigma > n | I_0 = k\right] = \left(\sum_{i=i_{\epsilon_1}+1}^{\infty} + \sum_{i=1}^{i_{\epsilon_1}}\right) q_i^{(k)} \left(1 - p_i^{(k)}\right)^n$$
$$\leq \sum_{i=i_{\epsilon_1}+1}^{\infty} q_i^{(k)} e^{-n p_i^{(k)}} + \left(1 - p_{i_{\epsilon_1}}^{(k)}\right)^n, \tag{62}$$

where the second inequality uses the monotonicity of $p_i^{(k)}$ and $1-x \leq e^{-x}$. Using a similar approach as in (52), we can upper bound (62) by

$$\mathbb{P}\left[\sigma > n | I_0 = k\right] \leq \left(\sum_{i=i_{\epsilon_1}+1}^{i_{\lfloor \log n\epsilon_2 \rfloor}-1} + \sum_{i=i_{\lfloor \log n\epsilon_2 \rfloor}}^{i_m} + \sum_{i=i_m+1}^{\infty}\right) q_i^{(k)} e^{-n p_i^{(k)}} + \left(1 - p_{i_{\epsilon_1}}^{(k)}\right)^n$$
$$\leq e^{-n\epsilon_2} + \sum_{l=m}^{\infty} e^{-e^l} \sum_{j=i_{l+1}+1}^{i_l} q_j^{(k)} + \sum_{i=i_m+1}^{\infty} q_i^{(k)} + \left(1 - p_{i_{\epsilon_1}}^{(k)}\right)^n$$
$$\leq e^{-n\epsilon_2} + \sum_{l=m}^{\infty} e^{-e^l} \frac{(1+\epsilon_1)}{\Phi_k(n/e^{l+1})} + \frac{1+\epsilon_1}{\Phi_k(n/e^m)} + \left(1 - p_{i_{\epsilon_1}}^{(k)}\right)^n,$$

which implies

$$\mathbb{P}\left[\sigma > n | I_0 = k\right] \Phi_k(n) \leq \frac{(1+\epsilon_1)\Phi_k(n)}{\Phi_k(n/e^m)} + \sum_{l=m}^{\infty} e^{-e^l} \frac{(1+\epsilon_1)\Phi_k(n)}{\Phi_k(n/e^{l+1})} + o(1). \tag{63}$$

Passing $\epsilon_1 \to 0$, $n \to \infty$ and then $m \to \infty$ in (63) yields

$$\overline{\lim}_{n\to\infty} \mathbb{P}\left[\sigma > n | I_0 = k\right] \Phi_k(n) \leq 1. \tag{64}$$

Combining (61) and (64) finishes the proof of (60).

Up to now, we have proved (47) for $\beta_k > 0$ and $\beta_k = 0$. Next, we use the concentration bounds (44) and (45) for $M(x)$ in Lemma 7.1 to characterize $M^{\leftarrow}(x)$.

Step 2. For $x_1 = m^{\leftarrow}(x/(1+\epsilon(x)))$, we obtain, by (44),

$$\begin{aligned}\mathbb{P}[M^{\leftarrow}(x) < x_1] &\leq \mathbb{P}\left[M(m^{\leftarrow}(x/(1+\epsilon(x)))) \geq x\right] \\ &= \mathbb{P}\left[M\left(m^{\leftarrow}\left(\frac{x}{1+\epsilon(x)}\right)\right) \geq \left(\frac{(1+\epsilon(x))x}{1+\epsilon(x)}\right)\right]. \end{aligned} \tag{65}$$

Recalling h_1 and h_2 defined in (14) and noting $\delta(x) \leq 1$, we have, for $x > x_0$, $\epsilon(x) \geq h_1\epsilon\left(x/(1+\epsilon(x))\right)$, which, in conjunction with (65) and using (44), implies that $\mathbb{P}[M^{\leftarrow}(x) < x_1]$ is upper bounded by

$$\begin{aligned}\mathbb{P}\left[M\left(x_1\right) \geq \left(1+h_1\epsilon\left(\frac{x}{1+\epsilon(x)}\right)\right)\left(\frac{x}{1+\epsilon(x)}\right)\right] &\leq \exp\left(-(h_1\epsilon(x/(1+\epsilon(x))))^2 x/\left(4\bar{s}(1+\epsilon(x))\right)\right) \\ &\leq \exp\left(-\frac{h_1^2}{h_2^2}\frac{\epsilon(x)^2 x}{4\bar{s}(1+\epsilon)}\right). \end{aligned} \tag{66}$$

Thus, by (46), (47), (44) and (66), we obtain

$$\begin{aligned}\mathbb{P}[C_0 > x] &\leq \mathbb{P}[\sigma > M^{\leftarrow}(x), M^{\leftarrow}(x) \geq x_1] + \mathbb{P}[M^{\leftarrow}(x) < x_1] \\ &\leq \mathbb{P}[\sigma > m^{\leftarrow}(x/(1+\epsilon(x)))] + \mathbb{P}[M^{\leftarrow}(x) < x_1] \\ &\lesssim \frac{\Gamma(\beta_k+1)}{\Phi_k\left(m^{\leftarrow}(x/(1+\epsilon(x)))\right)} + \exp\left(-\frac{h_1^2}{h_2^2}\frac{\epsilon(x)^2 x}{4\bar{s}(1+\epsilon)}\right). \end{aligned}$$

Using $\overline{\lim}_{x\to\infty} \log\left(m^{\leftarrow}(x)\right)/(\delta^2(x)x) = 0$ and (13), we obtain, recalling $\epsilon(x) = \epsilon\delta(x)$ and passing $\epsilon \to 0$,

$$\mathbb{P}[C_0 > x] \leq \frac{\Gamma\left(1+\beta_k\right)}{\Phi_k\left(m^{\leftarrow}(x)\right)} + o\left(1/\Phi_k\left(m^{\leftarrow}(x)\right)\right). \tag{67}$$

Let $x_2 = m^{\leftarrow}(x/(1-\epsilon(x)))$. We obtain

$$\mathbb{P}[C_0 > x] \geq \mathbb{P}[\sigma > M^{\leftarrow}(x), M^{\leftarrow}(x) \leq x_2] - \mathbb{P}[M^{\leftarrow}(x) > x_2],$$

which, by similar arguments as in proving (67), yields

$$\mathbb{P}[C_0 > x] \geq \frac{\Gamma\left(1+\beta_k\right)}{\Phi_k\left(m^{\leftarrow}(x)\right)} - o\left(1/\Phi_k\left(m^{\leftarrow}(x)\right)\right). \tag{68}$$

Combining (67) and (68) finishes the proof. □

7.3 Proof of Corollary 3.2

Consider $p_x^{(k)} \sim l(x)/x^{\alpha}$ with $l(x)$ being a slowly varying function. According to Proposition 1.5.10 of [20], we have

$$\mathbb{P}\left[R_0 > x\right] = \sum_{i\geq x} p_i^{(k)} \sim \int_x^{\infty} \frac{l(x)}{x^{\alpha}}dx \sim \frac{l(x)}{(\alpha-1)x^{\alpha-1}}. \tag{69}$$

Using Lemma 3.4, we obtain

$$m(z) \sim \sum_{i\geq 1}\left(1-\exp\left(-\frac{l(i)z}{i^{\alpha}}\right)\right) \sim \int_1^{\infty}\left(1-\exp\left(-\frac{l(x)z}{x^{\alpha}}\right)\right)dx.$$

Since $l(x)x^{-\alpha} \sim \alpha\int_x^{\infty} l(t)t^{-\alpha-1}dt$ (Proposition 1.5.10 of [20]), for any $\epsilon > 0$, there exists $x_{\epsilon} > 0$, such that for all $x > x_{\epsilon}$,

$$(1-\epsilon)\alpha\int_x^{\infty} l(t)t^{-\alpha-1} < l(x)x^{-\alpha} < (1+\epsilon)\alpha\int_x^{\infty} l(t)t^{-\alpha-1}dt. \tag{70}$$

Therefore, $m(z)$ can be upper bounded by

$$m(z) \lesssim \int_1^{x_\epsilon} \left(1 - \exp\left(-\frac{l(x)z}{x^\alpha}\right)\right) dx + \int_{x_\epsilon}^{\infty} \left(1 - \exp\left(-(1+\epsilon)\alpha z \int_x^\infty l(t)t^{-\alpha-1}dt\right)\right) dx.$$

Define $f(x) = \alpha \int_x^\infty l(t)t^{-\alpha-1}dt$. We obtain

$$\begin{aligned} m(z) &\lesssim x_\epsilon + \int_{x_\epsilon}^{\infty} \left(1 - e^{-(1+\epsilon)zf(x)}\right) dx \\ &= x_\epsilon + x\left(1 - e^{-(1+\epsilon)zf(x)}\right)\Big|_{x_\epsilon}^{\infty} + \int_{x_\epsilon}^{\infty} x de^{-(1+\epsilon)zf(x)} \\ &= x_\epsilon^{-(1+\epsilon)zf(x_\epsilon)} + \int_{x_\epsilon}^{\infty} x de^{-(1+\epsilon)zf(x)} \triangleq I_1 + I_2. \end{aligned} \tag{71}$$

For $y = f(x)$, we have

$$I_2 = \int_{x_\epsilon}^{\infty} x e^{-(1+\epsilon)zf(x)}(-(1+\epsilon)zf'(x))dx = \int_0^{f(x_\epsilon)} (1+\epsilon)zf^{\leftarrow}(y)e^{-(1+\epsilon)zy}dy, \tag{72}$$

where $f^{\leftarrow}$ is the inverse function of f. Let $g(x) = 1/x$, $h(x) = g \circ f(x)$, and $l_1(x) = l(x)^{-1/\alpha}$. We have $h(x) \sim x^\alpha {l_1}^\alpha(x)$. By Proposition 1.5.15 of [20], we obtain the asymptotic inverse of h,

$$h^{\leftarrow}(x) \sim x^{1/\alpha} l_1^{\#}(x^{1/\alpha}).$$

Recall $l_{n+1}(x) \triangleq l_1(x/l_n(x)), n = 1, 2, \ldots$ and $l_n(x) \sim l_{n+1}(x)$ as $x \to \infty$ for some $n \geq 2$. Using Proposition 2.3.5 of [20], we have $l_1^{\#} \sim 1/l_n(x)$. Therefore,

$$h^{\leftarrow}(x) \sim x^{1/\alpha}/l_n(x^{1/\alpha}).$$

Since $h = g \circ f$, we have

$$h^{\leftarrow}(x) = f^{\leftarrow}(g^{\leftarrow}(x)) = f^{\leftarrow}(1/x).$$

implying, as $x \to 0$,

$$f^{\leftarrow}(x) = h^{\leftarrow}(1/x) \sim \frac{1}{x^{1/\alpha} l_n(x^{-1/\alpha})}.$$

For $y = f(x_\epsilon)$ small enough, we have

$$\frac{1-\epsilon}{y^{1/\alpha} l_n(y^{-1/\alpha})} < f^{\leftarrow}(y) < \frac{1+\epsilon}{y^{1/\alpha} l_n(y^{-1/\alpha})}. \tag{73}$$

Combining (72) and (73) yields

$$I_2 < (1+\epsilon)^2 z \int_0^{f(x_\epsilon)} \frac{e^{-zy}}{y^{1/\alpha} l_n(y^{-1/\alpha})} dy < (1+\epsilon)^2 z \int_0^{\infty} \frac{e^{-zy}}{y^{1/\alpha} l_n(y^{-1/\alpha})} dy.$$

Using Theorem 1.7.1′ of [20], we obtain, as $z \to \infty$,

$$z \int_0^{\infty} \frac{e^{-zy}}{y^{1/\alpha} l_n(y^{-1/\alpha})} dy \sim \Gamma(1-1/\alpha) z^{1/\alpha}/l_n(z^{1/\alpha}), \tag{74}$$

implying

$$I_2 < (1+\epsilon)^3 \Gamma(1-1/\alpha) z^{1/\alpha}/l_n(z^{1/\alpha}).$$

Therefore, using (71), we have, for z large enough,

$$m(z) \lesssim (1+\epsilon)^3 \Gamma(1-1/\alpha) z^{1/\alpha}/l_n(z^{1/\alpha}) + x_\epsilon^{-(1+\epsilon)zf(x_\epsilon)}. \tag{75}$$

Next, we prove a lower bound for $m(z)$. Recalling (70) and using a similar approach as in (71), we have

$$\begin{aligned} m(z) &\gtrsim \int_{x_\epsilon}^{\infty} \left(1 - e^{-(1-\epsilon)zf(x)}\right) dx = x\left(1 - e^{-(1-\epsilon)zf(x)}\right)\Big|_{x_\epsilon}^{\infty} + \int_{x_\epsilon}^{\infty} x de^{-(1-\epsilon)zf(x)} \\ &> \int_0^{f(x_\epsilon)} (1-\epsilon)zf^{\leftarrow}(y)e^{-(1-\epsilon)zy}dy = \left(\int_0^{\infty} - \int_{f(x_\epsilon)}^{\infty}\right)(1-\epsilon)zf^{\leftarrow}(y)e^{-(1-\epsilon)zy}dy \\ &> \int_0^{\infty} (1-\epsilon)zf^{\leftarrow}(y)e^{-(1-\epsilon)zy}dy - x_\epsilon e^{-(1-\epsilon)zf(x_\epsilon)}. \end{aligned}$$

Using (73) and (74), we obtain

$$\begin{aligned} m(z) &\gtrsim (1-\epsilon)^2 z \int_0^{f(x_\epsilon)} \frac{e^{-zy}}{y^{1/\alpha} l_n(y^{-1/\alpha})} dy - x_\epsilon e^{-(1-\epsilon)zf(x_\epsilon)} \\ &\gtrsim (1-\epsilon)^3 \Gamma(1-1/\alpha) z^{1/\alpha}/l_n(z^{1/\alpha}) - x_\epsilon e^{-(1-\epsilon)zf(x_\epsilon)}. \end{aligned} \tag{76}$$

Combining (75) and (76), and passing $z \to \infty$ and then $\epsilon \to 0$, we obtain

$$m(z) \sim \Gamma(1-1/\alpha) z^{1/\alpha}/l_n(z^{1/\alpha}). \tag{77}$$

Define

$$F(z) = \frac{(\alpha-1)z^{\alpha-1}}{\Gamma(1-1/\alpha)^{\alpha-1} l(z)}.$$

Now, we show $F(z) \sim \Phi_k(m^{\leftarrow}(z))$ as $z \to \infty$, which is equivalent to $F(m(x^\alpha/l(x))) \sim \Phi(x^\alpha/l(x))$ as $x \to \infty$. Using (77), we obtain

$$\begin{aligned} F(m(x^\alpha/l(x))) &= \frac{\alpha-1}{\Gamma(1-1/\alpha)^{\alpha-1}} \frac{m(x^\alpha/l(x))^{\alpha-1}}{l(m(x^\alpha/l(x)))} \sim \frac{(\alpha-1)x^{\alpha-1} l_1(x)^{\alpha-1}}{l_n(xl_1(x))^{\alpha-1}} \Big/ l\left(\frac{\Gamma(1-1/\alpha)xl_1(x)}{l_n(xl_1(x))}\right) \\ &\sim \frac{(\alpha-1)x^{\alpha-1}c(x)^{\alpha-1}}{l(xc(x))}, \end{aligned} \tag{78}$$

where $c(x) = l_1(x)/l_n(xl_1(x))$.

Recall $l_n(y) \sim l_{n+1}(y) = l_1(y/l_n(y)), y \to \infty$. For $y = xl_1(x)$, by Proposition 1.5.15 of [20], we obtain

$$x \sim y l_1^{\#}(y) \sim y/l_n(y),$$

which implies

$$l_n(xl_1(x)) = l_n(y) \sim l_1(y/l_n(y)) \sim l_1(x).$$

Therefore, we obtain

$$\lim_{x\to\infty} c(x) = \lim_{x\to\infty} \frac{l_1(x)}{l_n(xl_1(x))} = 1. \tag{79}$$

Combining (4), (78) and (79) yields

$$\Phi(x^\alpha/l(x)) \sim F(m(x^\alpha/l(x))),$$

which implies

$$\Phi_k(m^{\leftarrow}(z)) \sim F(z) = \frac{(\alpha-1)z^{\alpha-1}}{\Gamma(1-1/\alpha)^{\alpha-1} l(z)}, \text{ as } z \to \infty.$$

Therefore, by Theorem 3.1, we obtain, as $x \to \infty$,

$$\mathbb{P}[C_0 > x] \sim \frac{\Gamma(2-1/\alpha)\Gamma(1-1/\alpha)^{\alpha-1}}{\alpha-1}\frac{l(x)}{x^{\alpha-1}}. \tag{80}$$

Combining (69) and (80) finishes the proof.

7.4 Proof of Theorem 3.6

The characteristic time approximation gives, for a LRU cache of size x,

$$\mathbb{P}_{CT}[C_0 > x | I_0 = k] = \sum_{i=1}^{\infty} q_i^{(k)} e^{-p_i^{(k)} T}, \tag{81}$$

where T is the unique solution to $\sum_{i=1}^{\infty}(1 - e^{-p_i^{\circ} T}) = x$. Using (8), (47) and $e^{-y} \le 1 - y$, we derive a lower bound of (81),

$$\begin{aligned}\mathbb{P}_{CT}[C_0 > x | I_0 = k] &\ge \sum_{i=1}^{\infty} q_i^{(k)} \left(1 - p_i^{(k)}\right)^T \\ &= \mathbb{P}[\sigma > T | I_0 = k] \sim \frac{\Gamma(\beta_k + 1)}{\Phi_k(T)}.\end{aligned} \tag{82}$$

Next, we derive an upper bound. Using a similar approach that proves an upper bound for $\mathbb{P}[\sigma > n | I_0 = k]$ in the proof of Theorem 3.1, we have, for ϵ_1 is defined in (50),

$$\begin{aligned}\mathbb{P}_{CT}[C_0 > x | I_0 = k] &= \sum_{i=1}^{i_{\epsilon_1}} q_i^{(k)} e^{-p_i^{(k)} T} + \sum_{i=i_{\epsilon_1}+1}^{\infty} q_i^{(k)} e^{-p_i^{(k)} T} \\ &\le e^{-p_{i_{\epsilon_1}}^{(k)} T} + \sum_{i=i_{\epsilon_1}+1}^{\infty} q_i^{(k)} e^{-p_i^{(k)} T} \lesssim \frac{\Gamma(\beta_k + 1)}{\Phi_k(T)}.\end{aligned} \tag{83}$$

Combining (82) and (83) yields, as $x \to \infty$,

$$\mathbb{P}_{CT}[C_0 > x | I_0 = k] \sim \frac{\Gamma(\beta_k + 1)}{\Phi_k(T)}. \tag{84}$$

By Lemma 3.4, we have $T \sim m^{\leftarrow}(x)$, which implies (81), by using the fact $\lim_{x\to\infty} x^{\beta_k} l_k(x)/\Phi_k(x) = 1$ and (84).

REFERENCES

[1] [n. d.]. Memcached. ([n. d.]). http://memcached.org/

[2] Susanne Albers and Jeffery Westbrook. 1998. Self-organizing data structures. *Online Algorithms: The state of the art* 1442 (1998), 13–41.

[3] Brian Allen and Ian Munro. 1978. Self-organizing binary search trees. *J. ACM* 25 (1978), 526–535.

[4] Noga Alon and Joel H. Spencer. 2000. *The probabilistic method* (2nd ed.). John Wiley.

[5] Sigal Ar, Bernard Chazelle, and Ayellet Tal. 2000. Self-customized BSP trees for collision detection. *Computational Geometry: Theory and Applications* 10 (2000), 23–29.

[6] Martin Arlitt, Rich Friedrich, and Tai Jin. 1999. Workload characterization of a Web proxy in a cable modem environment. *SIGMETRICS Performance Evalation Review* 27, 2 (Sept. 1999), 25–36.

[7] Martin Arlitt and Carey Williamson. 1997. Internet Web servers: workload characterization and performance implications. In *IEEE/ACM Transaction on Networking*.

[8] Sanjeev Arora, David Karger, and Marek Karpinski. 1995. Polynomial time approximation schemes for dense instances of NP-hard problems. In *Proceedings of the 27th STOC*. 284–293.

[9] Berk Atikoglu, Yuehai Xu, Eitan Frachtenberg, Song Jiang, and Mike Paleczny. 2012. Workload analysis of a large-scale key-value store. *SIGMETRICS Perform. Eval. Rev.* 40, 1 (June 2012), 53–64.

[10] Oleg I. Aven, Leonid B. Boguslavsky, and Yakov A. Kogan. 1976. Some results on distribution-free analysis of paging algorithms. *IEEE Trans. Computers* 25, 7 (1976), 737–745.
[11] Oleg Ivanovich Aven, Edward Grady Coffman, and Yakov Afroimovich Kogan. 1987. *Stochastic analysis of computer storage*. Vol. 38. Springer Science & Business Media.
[12] Ozalp Babaoglu and Domenico Ferrari. 1983. Two-level replacement decisions in paging. *IEEE Trans. Comput.* 100, 32 (1983).
[13] Pual Barford, Azer Bestavros, Aadam Bradley, and Mark Crovella. 1999. Changes in web client access patterns. In *World Wide Web Journal, Special Issue on Characterization and Performance Evaluation.*
[14] Javiera Barrera and Christian Paroissin. 2004. On the distribution of the search cost for the move-to-front rule with random weights. *Journal of Applied Probability* 41, 1 (03 2004), 250–262.
[15] Steven Bell and Ruth Williams. 2001. Dynamic scheduling of a system with two parallel servers in heavy traffic with resource pooling: asymptotic optimality of a threshold policy. *The Annals of Applied Probability* 11, 3 (08 2001), 608–649.
[16] Jon L. Bentley and Catherine C. McGeoch. 1985. Amortized analysis of self-organizing sequential search heuristics. *Commun. ACM* 28 (1985), 404–411.
[17] Daniel S. Berger, Philipp Gland, Sahil Singla, and Florin Ciucu. 2014. Exact analysis of TTL cache networks. *Performance Evaluation* 79 (2014), 2–23.
[18] Daniel S. Berger, Sebastian Henningsen, Florin Ciucu, and Jens B. Schmitt. 2015. Maximizing cache hit ratios by variance reduction. *SIGMETRICS Performance Evaluation Review* 43, 2 (Sept. 2015), 57–59.
[19] Christian Berthet. 2017. Approximation of LRU caches miss rate: application to power-law popularities. *arXiv:1705.10738* (2017).
[20] N. H. Bingham, C. M. Goldie, and J. L. Teugels. 1987. *Regular Variation.* Cambridge University Press.
[21] James R. Bitner. 1979. Heuristics that dynamically organize data structures. *SIAM J. Comput.* 8 (1979), 82–110.
[22] Avrim Blum, Shuchi Chawla, and Adam Kalai. 2002. Static optimality and dynamic search-optimality in lists and trees. In *Proceedings of the 13th SODA*. 1–8.
[23] Christian Borgs, Jennifer T. Chayes, Sherwin Doroudi, Mor Harchol-Balter, and Kuang Xu. 2013. The optimal admission threshold in observable queues with state dependent pricing. *Probability in the Engineering and Informational Sciences* 28, 1 (13 12 2013), 101–119.
[24] Allan Borodin and Ran El-Yaniv. 1998. *Online Computation and Competitive Analysis.* Cambridge University Press, New York, NY, USA.
[25] Allan Borodin, Prabhakar Raghavan, Sandy Irani, and Baruch Schieber. 1991. Competitive paging with locality of reference. In *Proceedings of the Twenty-third Annual ACM Symposium on Theory of Computing (STOC '91)*. 249–259.
[26] Lee Breslau, Pei Cao, Li Fan, Graham Phillips, and Scott Shenker. 1999. Web caching and Zipf-like distributions: evidence and implications. In *Proceedings of the 18th Conference on Information Communications.*
[27] Nathan Bronson, Zach Amsden, George Cabrera, Prasad Chakka, Peter Dimov, Hui Ding, Jack Ferris, Anthony Giardullo, Sachin Kulkarni, Harry Li, Mark Marchukov, Dmitri Petrov, Lovro Puzar, Yee Jiun Song, and Venkat Venkataramani. 2013. TAO: Facebook's distributed data store for the social graph. In *Proceedings of the 2013 USENIX Conference on Annual Technical Conference (USENIX ATC'13)*. USENIX Association, Berkeley, CA, USA, 49–60.
[28] Hao Che, Ye Tung, and Zhijun Wang. 2002. Hierarchical Web caching systems: modeling, design and experimental results. *IEEE Journal on Selected Areas in Communications* 20, 7 (Sep 2002), 1305–1314.
[29] Weibo Chu, Mostafa Dehghan, Don Towsley, and Zhi-Li Zhang. 2016. On allocating cache resources to content providers. In *Proceedings of the 2016 conference on 3rd ACM Conference on Information-Centric Networking*. ACM, 154–159.
[30] F R Chung, D J Hajela, and P D Seymour. 1985. Self-organizing sequential search and Hilbert's inequalities. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing (STOC '85)*. 217–223.
[31] Asaf Cidon, Assaf Eisenman, Mohammad Alizadeh, and Sachin Katti. 2016. Cliffhanger: Scaling performance cliffs in web memory caches. In *USENIX NSDI*. 379–392.
[32] Edward G. Coffman, Jr. and Peter J. Denning. 1973. *Operating Systems Theory.* Prentice Hall Professional Technical Reference.
[33] Carlos Cunha, Azer Bestavros, and Mark Crovella. 1995. *Characteristics of WWW client-based traces.* Technical Report. Boston University, Boston, MA, USA.
[34] Mostafa Dehghan, Weibo Chu, Philippe Nain, and Don Towsley. 2017. Sharing LRU cache resources among content providers: a utility-based approach. *arXiv preprint arXiv:1702.01823* (2017).
[35] Robert P. Dobrow and James Allen Fill. 1995. The move-to-front rule for self-organizing lists with Markov dependent requests. In *Discrete Probability and Algorithms*. Springer New York, New York, NY, 57–80.
[36] Bradley M. Duska, David Marwood, and Michael J. Feely. 1999. The measured access characteristics of world-wide-web client proxy caches. In *Proceedings of the USENIX Symposium on Internet Technologies and Systems (USITS '97).*

[37] Vladmir Estivill-Castro and Derick Wood. 1992. A survey of adaptive sorting algorithms. *Comput. Surveys* 24 (1992), 441–476.

[38] Ronald Fagin. 1977. Asymptotic miss ratios over independent references. *J. Comput. System Sci.* 14, 2 (1977), 222–250.

[39] Bin Fan, David G. Andersen, and Michael Kaminsky. 2013. MemC3: Compact and concurrent MemCache with dumber caching and smarter hashing. In *Proceedings of the 10th USENIX Conference on Networked Systems Design and Implementation (NSDI'13)*. USENIX Association, Lombard, IL, 371–384.

[40] Andrés Ferragut, Ismael Rodriguez, and Fernando Paganini. 2016. Optimizing TTL caches under heavy-tailed demands. *SIGMETRICS Perform. Eval. Rev.* 44, 1 (June 2016), 101–112.

[41] JamesAllen Fill. 1996. An exact formula for the move-to-front rule for self-organizing lists. *Journal of Theoretical Probability* 9, 1 (1996), 113–160.

[42] James Allen Fill. 1996. Limits and rates of convergence for the distribution of search cost under the move-to-front rule. *Theoretical Computer Science* 164, 1 (1996), 185 – 206.

[43] Philippe Flajolet, Danièle Gardy, and Loÿs Thimonier. 1992. Birthday paradox, coupon collectors, caching algorithms and self-organizing search. *Discrete Applied Mathematics* 39, 3 (November 1992), 207–229.

[44] N. Choungmo Fofack, Philippe Nain, Giovanni Neglia, and Don Towsley. 2012. Analysis of TTL-based cache networks. In *Performance Evaluation Methodologies and Tools (VALUETOOLS), 2012 6th International Conference on*. IEEE, 1–10.

[45] Nicaise Choungmo Fofack, Philippe Nain, Giovanni Neglia, and Don Towsley. 2014. Performance evaluation of hierarchical TTL-based cache networks. *Computer Networks* 65 (2014), 212 – 231.

[46] Christine Fricker, Philippe Robert, and James Roberts. 2012. A versatile and accurate approximation for LRU cache performance. In *Proceedings of the 24th International Teletraffic Congress (ITC '12)*. Article 8, 8 pages.

[47] Massimo Gallo, Bruno Kauffmann, Luca Muscariello, Alain Simonian, and Christian Tanguy. 2014. Performance evaluation of the random replacement policy for networks of caches. *Performance Evaluation* 72 (2014), 16–36.

[48] Michele Garetto, Emilio Leonardi, and Valentina Martina. 2016. A unified approach to the performance analysis of caching systems. *ACM Transactions on Modeling and Performance Evaluation of Computing Systems* 1, 3 (2016), 12.

[49] Nicolas Gast and Benny Van Houdt. 2015. Transient and steady-state regime of a family of list-based cache replacement algorithms. In *Proceedings of the 2015 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '15)*. 123–136.

[50] Erol Gelenbe. 1973. A unified approach to the evaluation of a class of replacement algorithms. *IEEE Trans. Comput.* 100, 6 (1973), 611–618.

[51] Gaston H. Gonnet, J.Ian Munro, and Hendra Suwanda. 1981. Exegesis of self-organizing linear search. *SIAM J. Comput.* 10 (1981), 613–637.

[52] J. Michael Harrison and Marcel J. López. 1999. Heavy traffic resource pooling in parallel server systems. *Queueing Systems* 33, 4 (1999), 339–368.

[53] J.H. Hester and D.S. Hirschberg. 1985. Self-organizing linear search. *Comput. Surveys* 17 (1985), 295–311.

[54] Ryo Hirade and Takayuki Osogami. 2010. Analysis of page replacement policies in the fluid limit. *Operations research* 58, 4-part-1 (2010), 971–984.

[55] Sandy Irani, Anna R. Karlin, and Steven Phillips. 1992. Strongly competitive algorithms for paging with locality of reference. In *Proceedings of the Third Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '92)*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 228–236.

[56] Predrag R. Jelenković. 1999. Asymptotic approximation of the move-to-front search cost distribution and least-recently-used caching fault probabilities. *The Annals of Applied Probability* 2 (1999), 430–464.

[57] P. R. Jelenković. 2002. Least-recently-used caching with Zipf's law requests. In *The Sixth INFORMS Telecommunications Conference*. Boca Raton, Florida.

[58] Predrag R Jelenković and Xiaozhu Kang. 2007. LRU caching with moderately heavy request distributions. In *2007 Proceedings of the Fourth Workshop on Analytic Algorithmics and Combinatorics (ANALCO)*. SIAM, 212–222.

[59] Predrag R. Jelenković and Ana Radovanović. 2004. Least-recently-used caching with dependent requests. *Theoretical Computer Science* 326, 1-3 (Oct. 2004), 293–327.

[60] Song Jiang and Xiaodong Zhang. 2002. LIRS: An efficient low inter-reference recency set replacement policy to improve buffer cache performance. In *Proceedings of the 2002 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '02)*. ACM, New York, NY, USA, 31–42.

[61] Jaeyeon Jung, Arthur W. Berger, and Hari Balakrishnan. 2003. Modeling TTL-based Internet Caches. In *IEEE Infocom 2003*. San Francisco, CA.

[62] Offer Kella. 1989. The threshold policy in the M/G/1 queue with server vacations. *Naval Research Logistics (NRL)* 36, 1 (1989), 111–123.

[63] Donald E. Knuth. 1998. *The Art of Computer Programming, Volume 3: (2Nd Ed.) Sorting and Searching*. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA.

[64] Nikolaos Laoutaris, Hao Che, and Ioannis Stavrakakis. 2006. The LCD interconnection of LRU caches and its analysis. *Performance Evaluation* 63, 7 (2006), 609–634.
[65] Stephen Manley and Margo Seltzer. 1997. Web facts and fantasy. In *Proceedings of the USENIX Symposium on Internet Technologies and Systems*.
[66] Nimrod Megiddo and Dharmendra S. Modha. 2003. ARC: a self-tuning, low overhead replacement cache. In *Proceedings of the 2Nd USENIX Conference on File and Storage Technologies (FAST '03)*. USENIX Association, Berkeley, CA, USA, 115–130.
[67] Rajeev Motwani. 1994. Average-case analysis of algorithm for matching and related problems. *J. ACM* 41 (1994), 1329–1356.
[68] Peter R Nelson. 1977. Single-shelf library-type Markov chains with infinitely many books. *Journal of Applied Probability* (1977), 298–308.
[69] Rajesh Nishtala, Hans Fugal, Steven Grimm, Marc Kwiatkowski, Herman Lee, Harry C. Li, Ryan McElroy, Mike Paleczny, Daniel Peek, Paul Saab, David Stafford, Tony Tung, and Venkateshwaran Venkataramani. 2013. Scaling Memcache at Facebook. In *Presented as part of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)*. USENIX, Lombard, IL, 385–398.
[70] Takayuki Osogami. 2010. A fluid limit for a cache algorithm with general request processes. *Advances in Applied Probability* 42, 3 (2010), 816–833.
[71] Stefan Podlipnig and Laszlo Böszörmenyi. 2003. A survey of web cache replacement strategies. *ACM Computing Surveys (CSUR)* 35, 4 (Dec. 2003), 374–398.
[72] Guocong Quan, Kaiyi Ji, and Jian Tan. 2018. LRU caching with dependent competing requests. In *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications (INFOCOM 2018)*. Honolulu, USA.
[73] Ronald Rivest. 1976. On self-organizing sequential search heuristics. *Commun. ACM* 19 (1976), 63–67.
[74] James Roberts and Nada Sbihi. 2013. Exploring the memory-bandwidth tradeoff in an information-centric network. In *Teletraffic Congress (ITC), 2013 25th International*. IEEE, 1–9.
[75] Liam Roditty and Uri Zwick. 2004. A fully dynamic reachability algorithm for directed graphs with an almost linear update time. In *Proceedings of the 36th STOC*. 184–191.
[76] Elisha J. Rosensweig, Jim Kurose, and Don Towsley. 2010. Approximate models for general cache networks. In *Proceedings of the 29th Conference on Information Communications (INFOCOM'10)*. IEEE Press, San Diego, California, USA, 1100–1108.
[77] Albers S. and M. Mitzenmacher. 1998. Average case analyses of list update algorithms. *Algorithmica* 21 (1998), 312–329.
[78] Daniel D. Sleator and Robert E. Tarjan. 1985. Amortized efficiency of list update and paging rules. *Commun. ACM* 28 (1985), 202–208.
[79] Daniel D. Sleator and Robert E. Tarjan. 1985. Self-adjusting binary search trees. *J. ACM* 32 (1985), 652–686.
[80] Alexander L. Stolyar. 2004. MaxWeight scheduling in a generalized switch: State space collapse and workload minimization in heavy traffic. *The Annals of Applied Probability* 14, 1 (02 2004), 1–53.
[81] Toyoaki Sugimoto and Naoto Miyoshi. 2006. On the asymptotics of fault probability in least-recently-used caching with Zipf-type request distribution. *Random Structures & Algorithms* 29, 3 (2006), 296–323.
[82] Jian Tan, Li Zhang, and Yandong Wang. 2015. Miss behavior for caching with lease. *SIGMETRICS Performance Evaluation Review, MAMA workshop* 43, 2 (2015), 60–62.
[83] Andrew S. Tanenbaum. 2001. *Modern Operating Systems* (2rd ed.). Prentice Hall Press, Upper Saddle River, NJ, USA.
[84] Naoki Tsukada, Ryo Hirade, and Naoto Miyoshi. 2012. Fluid limit analysis of FIFO and RR caching for independent reference models. *Performance Evaluation* 69, 9 (2012), 403–412.
[85] Jia Wang. 1999. A survey of Web caching schemes for the Internet. *SIGCOMM Computer Communication Review* 29, 5 (Oct. 1999), 36–46.
[86] Xingbo Wu, Li Zhang, Yandong Wang, Yufei Ren, Michel Hack, and Song Jiang. 2016. zExpander: a key-value cache with both high performance and fewer misses. In *Proceedings of the Eleventh European Conference on Computer Systems (EuroSys '16)*. ACM, New York, NY, USA, Article 14, 15 pages.
[87] Yuehai Xu, Eitan Frachtenberg, Song Jiang, and Mike Paleczny. 2014. Characterizing Facebook's Memcached workload. *IEEE Internet Computing* 18, 2 (2014), 41–49.
[88] Yuehai Xua, Eitan Frachtenbergb, and Song Jiang. 2014. Building a high-performance key-value cache as an energy-efficient appliance. *Performance Evaluation* 79 (September 2014), 24–37.
[89] Yue Yang and Jianwen Zhu. 2016. Write skew and Zipf distribution: Evidence and implications. *ACM Transactions on Storage (TOS)* 12, 4, Article 21 (June 2016), 19 pages.