Ephemeral Content Popularity at the Edge and Implications for On-Demand Caching

Niklas Carlsson and Derek Eager

Abstract—The ephemeral content popularity seen with many content delivery applications can make indiscriminate on-demand caching in edge networks highly inefficient, since many of the content items that are added to the cache will not be requested again from that network. In this paper, we address the problem of designing and evaluating more selective edge-network caching policies. The need for such policies is demonstrated through an analysis of a dataset recording YouTube video requests from users on an edge network over a 20-month period. We then develop a novel workload modelling approach for such applications and apply it to study the performance of alternative edge caching policies, including indiscriminate caching and *cache on* k^{th} *request* for different k. The latter policies are found able to greatly reduce the fraction of the requested items that are inserted into the cache, at the cost of only modest increases in cache miss rate. Finally, we quantify and explore the potential room for improvement from use of other possible predictors of further requests. We find that although room for substantial improvement exists when comparing performance to that of a perfect "oracle" policy, such improvements are unlikely to be achievable in practice.

Index Terms—Ephemeral content popularity; One-timers; One-hit-wonders; Edge network; Measurements; Caching

1 Introduction

Content delivery applications commonly rely on caching at servers distributed throughout the Internet, so as to achieve scalability and reduce access latency. Such an architecture can be highly efficient and effective for highly-popular content. In many content delivery applications, however, content popularity is usually short lived at best. There is a high rate of addition of new content, and for most of these content items there is only a brief period of time over which retrieval request(s) are received. Caching of such "ephemeral" content at an edge-network cache has little benefit relative to the associated costs of cache pollution, increased cache access delay owing to the load caused by content insertions, and (relevant for SSD-based caches) high write rates.

In this paper we address the problem of designing and evaluating edge-network caching policies for content delivery applications with ephemeral content. We first gain an understanding of the characteristics of such applications by analyzing a dataset recording YouTube requests from an edge network. We then propose a novel workload modelling approach for content delivery applications with ephemeral content, and apply it in evaluations of a class of simple edge-network caching policies. Finally, we investigate the potential for further caching policy improvements.

The paper makes the following primary contributions:

- We analyze YouTube request characteristics as observed at an edge network over a 20 month period, particularly with respect to ephemeral content. We observe that 71% of the requested videos are "one-timers" that are requested only once from the edge network during the 20 month observation period, demonstrating the need for selective caching policies.[1] Motivated by the observation that such policies should predict what *not* to cache, we take a closer look at one-timers and other videos receiving few views, and compare their characteristics with videos that receive more views.
- We propose a workload modelling approach suitable for content delivery applications with ephemeral content. Unlike most previous workload models, our model is not based on an assumption of a fixed collection of content items with stationary popularities. Instead, we assume that content items have finite lifetimes of interest, and model the distribution of the total number of times that a content item will be requested. Cache performance is then analyzed focusing on the tradeoff between the cache insertion rate, and the cache miss rate owing to items that have not previously been cached.
- We apply our workload model to the analysis of edge-network caching policies, specifically *cache on* k^{th} *request* policies for different k. We find that such policies can be highly effective at reducing the cache insertion rate at modest cost in increased cache miss rate.
- Finally, we explore the question of how much scope there may be for further cache performance improvements beyond those offered by *cache on* k^{th} *request* policies, for example by adopting popularity prediction methods based on content item meta data. Perhaps surprisingly, we find that although there is

- *N. Carlsson is with the Department of Computer and Information Science, Linköping University, Sweden. E-mail: niklas.carlsson@liu.se*
- *Derek Eager is with the Department of Computer Sciene, University of Saskatchewan, Canada.*

Manuscript received XXXX yy, 2015; accepted Sept. 2016

1. While the term "one-hit-wonders" recently was used to refer to web objects and videos only accessed once [26], we chose to use the shorter term "one-timers". However, the two terms are interchangeable.

substantial room for improvement when comparing to a perfect "oracle" policy, such improvements are unlikely to be possible in practice as they require accurate discrimination within the class of content items that will receive few, if any, future requests.

The reminder of the paper is organized as follows. Section 2 presents our data collection methodology and a high-level overview of our YouTube dataset. Section 3 takes a closer look at the distribution of the total number of observed requests for a video, ephemeral popularity, and the diversity in popularity characteristics. Sections 4 and 5 present our workload model and evaluate the performance tradeoffs seen by alternative edge caching policies. Finally, after discussing related work (Section 6), the paper concludes with a summary of our findings and directions for future work (Section 7).

2 Data Collection

2.1 Methodology

Longitudinal edge data: Our primary dataset is based on a trace logging all HTTP transactions at a large university campus. The trace spans 20 months and was collected between July 1, 2008, and February 28, 2010. First, a Bro [32][2] script summarizes all HTTP transactions on port 80 on the university's Internet link. The script captures both application-layer information (e.g., host name, URI, file type, status codes, etc.) and transport-layer information (e.g., transfer duration, bytes transferred, etc.). Second, through careful filtering, we extract all transactions associated with playback of YouTube videos. Each video is identified by parsing the YouTube URI. This step required some engineering and care, as the format of the URI varied over the duration of the measurement period, as well as in some cases also from device-to-device (depending on user agent). Third, we use a basic threshold-based filter that groups multiple transactions associated with playback of the same file and client into a single *aggregate* request. We say that two transactions are part of the same aggregate request if the transactions are done by the same client and the requests either overlap or are separated by less than some threshold time Δ (typically 5 minutes).[3] The aggregate video request is considered to have been placed at the time that the first such transaction was initiated by the client. While the results are relatively independent of the threshold value used, for completeness, Table 1 includes summary statistics for three different thresholds: 5 minutes, 30 minutes, and 2 hours. During the 20-month-long measurement campaign, the approximately 35,000 faculty, staff, and students of the university generated roughly 5.5 million aggregate requests to 2.4 million unique YouTube videos.

Server-side information: To support some of the analysis in this paper, a second data collection was undertaken at a later point in time, to obtain information from the YouTube site about each of the videos that were requested during our longitudinal measurement period. Three machines were used to scrape (at the pace allowed by the YouTube API) the YouTube meta data for the 2,392,688 videos. These measurement scrapes were performed between December 9, 2012, and January 25, 2013. The scrapes provided us with time-invariant information about the upload date/time, the duration of the video, the video category, and the uploader ID. The scrapes also provided us with the global view count at the time of the crawl; however, it is important to note that this view count may not reflect the view count at the time of the individual transactions of our longitudinal dataset. Although this second data collection took place well after the longitudinal collection was terminated, the crawls still provided information about 1,456,230 (60.86%) of the observed videos. Similar numbers have been observed by Islam et al. [23], who were able to observe 67.13% and 55.23% of random (recently uploaded) and popular (keyword search) videos, respectively, when revisiting the YouTube site looking for videos from a prior data collection that had been carried out more than two years earlier [6].

2. Bro Network Security Monitor, https://www.bro.org/

3. For privacy reasons, clients are anonymized on a daily basis, and we could therefore typically only group transactions on a per-day basis. Long-durations clusters of transactions that spanned across the 4:00AM boundary, at which time the client counters were reset, are therefore most often classified as two separate video requests, although the YouTube client ID allowed us to identify and repair a few such cases. To avoid affecting client privacy this was only done on a YouTube transaction basis.

TABLE 1
Summary statistics of the longitudinal local YouTube request workload.

Metric	Value
Unique videos	2,392,688
Viewing requests (5 min thresh)	5,501,398
Viewing requests (30 min thresh)	5,501,200
Viewing requests (2 hour thresh)	5,501,132
Start date	July 1, 2008
End date	Feb. 28, 2010

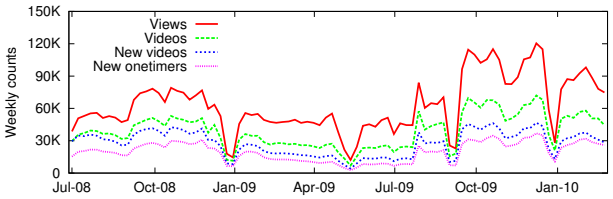
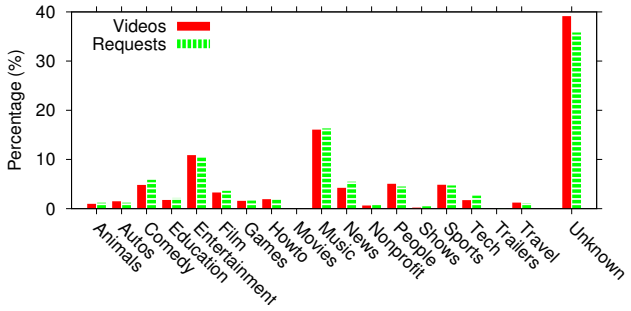


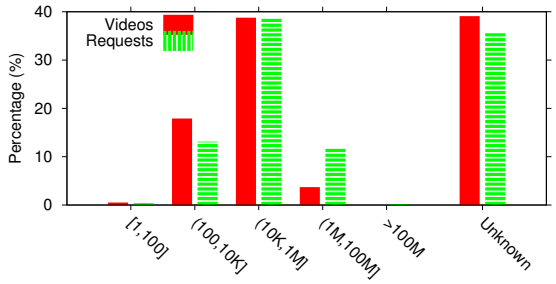
Fig. 1. Longitudinal weekly viewing pattern.

2.2 Basic Characteristics of Dataset

Figure 1 shows the weekly viewing pattern over the duration of our longitudinal data collection period. We make three observations. First, there is significant seasonal variations (e.g., there are significantly fewer views during holidays and there are more views during the fall terms than spring terms). Second, there is a general overall increase in the usage (e.g., comparing the two fall terms, or the two spring terms). Third, and perhaps most importantly, ignoring the first 3-4 months, the fraction of views to new videos (that have not been observed before in the trace) is roughly constant throughout the measurement period. Calculating statistics over all 86 weeks, on average about 70% of the videos viewed in a week had not been viewed by a campus user in any of the prior weeks. For the median week, 66% of the videos had not been viewed, while the week with the smallest fraction of new videos still has 50% new videos. These results show that neither the set of available content items or their item popularities are



(a) Category



(b) Popularity

Fig. 2. Breakdown of observed videos and viewing requests associated with each video category and global popularity class.

stationary over the measurement interval. Instead, many videos are viewed only one time ("one-timers"), or a small number of times.

To understand the edge-network workload, we also break down the videos and their viewing requests based on their video category (Figure 2(a)), and their global video popularity as measured by their total view count at the time of our server-side data collection (Figure 2(b)), when available. In total, out of the 2.4 million unique videos, the 39% of videos for which we were not able to obtain server-side information were responsible for 36% of the 5.5 million viewing requests. For the other videos, we found that music videos (16%) and videos that had between 10,000 and 1 million global views in December 2013 (39%) are the most popular, along these two classification dimensions, respectively. We also note that there is a relatively higher fraction of viewing requests to popular videos, compared to the fraction of popular videos observed in the trace. This is to be expected, as it simply implies that these globally popular videos also obtained relatively more views on the campus network.

3 Video Request Characterization

As observed in Section 2.2, the set of available content items is not fixed over the measurement interval, nor are content item popularities stationary. An important question when designing good caching policies, is therefore: when we see a request for a "new" video (such that there have been no requests observed yet from the edge network), what is the probability that it will get r additional requests from that edge network in the future? To answer this question, we calculate statistics for the number of requests per video.

TABLE 2
Power-law fitting summary.

Relationship	Scale parameter estimation	
	Scale parameter (α)	Standard error (σ_α)
Per-video basis	2.341	0.001
Per-request basis	1.4359	0.0003

3.1 Distribution of Number of Requests

Figure 3(a) shows the percentage of videos, and the percentage of viewing requests attributed to videos, with less than or equal to X viewing requests as Cumulative Distribution Functions (CDFs). We note that 71% of the total number of videos are one-timers and that these videos are responsible for 31% of the total viewing requests. In addition, there is a significant number of videos with only 2 or 3 viewing requests. The Complementary Cumulative Distribution Functions (CCDFs) shown in Figure 3(b) focus on the tail of the distribution. We note that both CCDFs show clear power-law characteristics, as indicated by the linear relationship when plotted on a log-log scale. To investigate this relationship, we performed power-law fitting on the data [12].

More specifically, to estimate the scale parameter (α) and the standard error (σ_α) of this estimation, we use the maximum likelihood estimator (MLE) for a discrete distribution with $x_{min} = 1$, and numerically solve the equation:

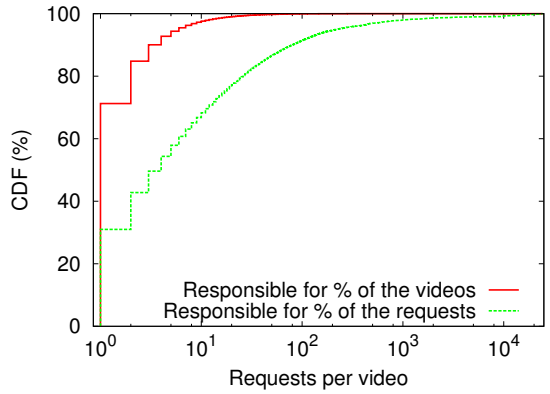
$$\frac{\zeta'(\alpha)}{\zeta(\alpha)} = -\frac{1}{n}\sum_{i=1}^{n} \ln x_i, \quad (1)$$

where x_i is the number of views to each video i, $1 \leq i \leq n$, and $\zeta(\alpha) = \sum_{i=1}^{\infty} i^{-\alpha}$ is the Riemann zeta function. (For details on the standard error calculations, we refer the interested reader to the work by Clauset et al. [12].) For completeness, we also estimate the scale parameter, when observations x_i are on a per-request basis, rather than on a per-video basis. Not surprisingly, with a relatively larger fraction of the requests being to videos with many requests, we observe a smaller scale parameter for this case. The scale parameter on a per-request basis is found to be approximately 1.44, while that on a per-video basis is approximately 2.34. Table 2 summarizes our results.

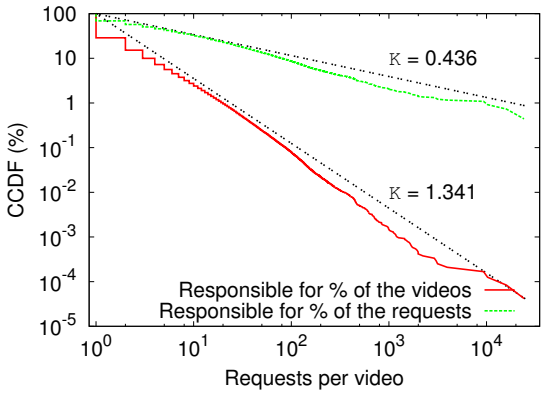
When interpreting these scale parameters it should be noted that there is a direct relationship between the Pareto distribution's shape parameter κ (roughly equal to the negative slope in the CCDF figure), and the power-law scale parameter α derived using the above MLE approach. In particular, the Pareto shape parameter (and slope in the CCDFs) $\kappa = \alpha - 1$ [27]. To illustrate this relationship, we have included the lines for $\kappa = 0.436$ and $\kappa = 1.34$ in Figure 3.

To understand the effect of finite trace duration, in addition to statistics for the full trace, we also calculated these summary statistics for newly-observed videos, as measured over shorter time durations. The distribution results are shown in Figure 4 for percentage of videos, with scale parameter estimations and their standard errors summarized in Table 3 for both per-video and per-request statistics.

For these traces we only counted statistics for the videos that were requested for the first time during the last year, the

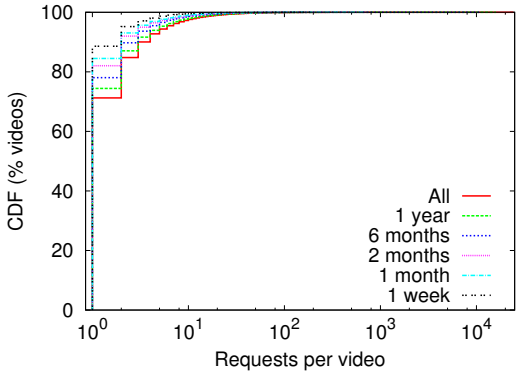


(a) CDF

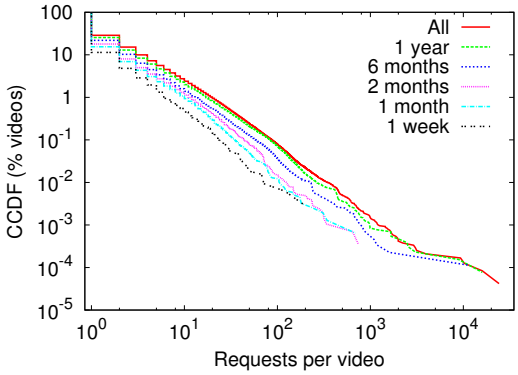


(b) CCDF

Fig. 3. Video popularity distribution, calculated as the percentage of videos and the percentage of viewing requests that were to videos with (a) less than or equal to X viewing requests, and (b) more than or equal to X viewing requests. Different scales are used to focus on the body and tail of the distributions, respectively.



(a) CDF (responsible for % videos)



(b) CCDF (responsible for % videos)

Fig. 4. File popularity distributions, as measured over different trace durations.

TABLE 3
Trace duration dependence in power-law fitting.

Time period	Scale parameter estimation	
	Video basis	Request basis
1 week	3.31 ± 0.02	2.036 ± 0.005
1 month	2.958 ± 0.006	1.773 ± 0.002
2 months	2.818 ± 0.004	1.693 ± 0.001
6 months	2.625 ± 0.002	1.5635 ± 0.0005
1 year	2.452 ± 0.001	1.4690 ± 0.0003
All	2.341 ± 0.001	1.4359 ± 0.0003

last 6 months, the last 2 months, the last month, or the last week of the full trace, respectively. Selecting to collect only statistics for new files corresponds to using a longer and longer warm-up period, and helps us avoid tagging a video request as a first time request due to the use of a finite trace duration. For example, in the case of the 1-year statistics, we have a warm-up period of 8 months.

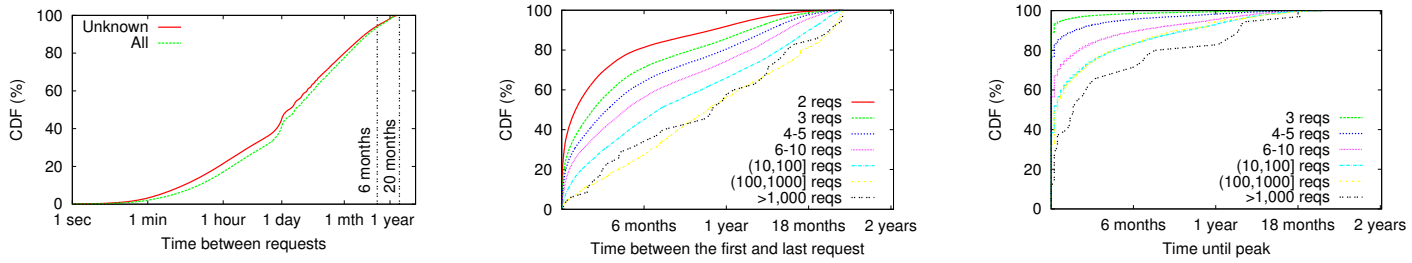
While the CCDFs and scale parameters are relatively insensitive to the trace duration once the traces become long enough, we note that the shorter duration traces have a somewhat higher proportion of one-timers, and correspondingly larger scaling parameter values. The higher proportion of one-timers in these traces is clearly seen in Figure 4(a). For example, the proportion of one-timers using the 6-month trace is 78%, compared to 71% when using the full trace.

3.2 Ephemeral Popularity

As discussed in Section 2.2 and shown by the large fraction of new videos each week (on average about 70% in Figure 1), there is a steady addition of new videos that obtain viewing requests. Furthermore, most videos are only either requested just once, or only a few times with most of the requests occurring relatively close to the first request.

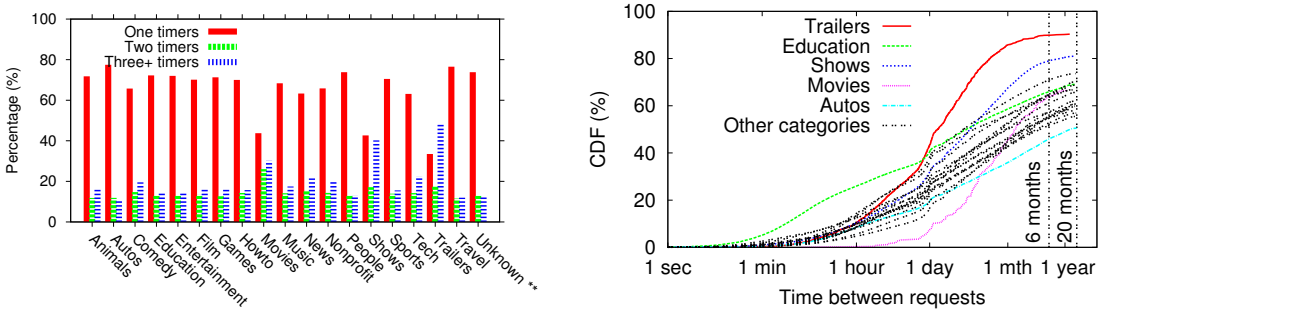
Figure 5(a) shows the CDF of inter-request times, for videos with at least 2 requests, while Figure 5(b) shows the CDF of the time between the first and last observed requests to a video, with videos categorized according to their total number of requests. Note that most videos with 10 or fewer requests, for example, experience all of their requests within 6 months of their first request. In contrast, if video popularities were stationary and requests not clustered, we would expect a typical video with 10 observed requests, for example, to have a time between its first to last request over the 20 month period of the trace of more than 16 months.

The frequently ephemeral nature of video popularity is also apparent when looking at the time until the week during which a video sees the most requests (with ties broken in favor of the earlier week), with CDF shown in Figure 5(c). Note that the peak request rate for most videos happens within the first few weeks of the first observed request. Even for relatively popular videos with between 100 and 1000 requests, more than 80% of these videos have their peak week within 6 months of their first request.



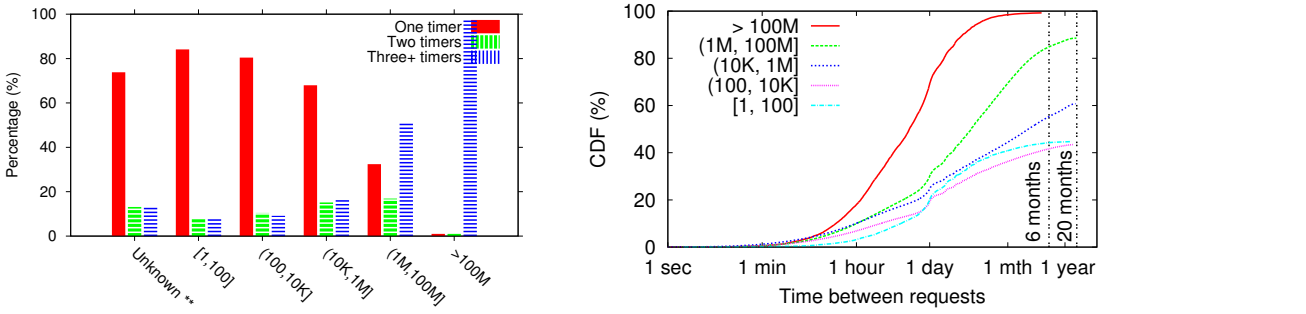
(a) Inter-request times (b) Time between the first and last request (c) Time until peak week

Fig. 5. Time spread of viewing requests for videos with different numbers of local requests.



(a) One-timer statistics (b) Inter-request time distributions

Fig. 6. Per-category breakdown of local video popularity.



(a) One-timer statistics (b) Inter-request time distributions

Fig. 7. Breakdown of local video popularity, based on global video popularity.

3.3 Diversity in Popularity Characteristics

Clearly there are significant differences in popularity between individual videos. We have also observed significant differences between different video categories. This is illustrated by the fraction of one-timers in each video category (Figure 6(a)) and the inter-request time distribution for videos in each category (Figure 6(b)). We note that "Movies", "Shows", and "Trailers" all have a smaller fraction of one-timers than the other categories. In contrast, "Autos" has the largest fraction of one-timers, perhaps suggesting that these videos have a relatively narrow niche viewership. The "Education" category stands out as it has the largest fraction of closely spaced requests. We speculate that these short inter-request times (e.g., more than 40% within a single day) may be due to videos shared by teachers, or among classmates at specific times.

Of course, some of these differences are also seen when looking at total view counts at the YouTube servers. Figure 7 provides some insight into the correlation between local on-campus and global popularity. Here, we show the fraction of one-timers (Figure 7(a)) and the inter-request times to videos with different global popularity (Figure 7(b)). These results confirm our intuition that there is a strong correlation. While there only are a few videos with more than 100M global views seen on campus (Figure 2(b)), almost all these see at least three views on campus (Figure 7(a)) and typically have very short inter-request times (e.g., roughly 70% of the inter-request times are less than one day; Figure 7(b)). Substantial correlation is also clearly evident for the videos with between 1M and 100M global views. For less popular videos (less than 1M global views), the majority (70-80%) of the observed videos are one-timers.

4 Cache Performance Model

4.1 Workload Model

Misses at an edge network cache can result from the action of a cache replacement policy (the requested content was earlier evicted from the cache), or can result from the requested content having never been present in the cache (either first-time reference at that edge network, or the content was previously referenced but not cached). We focus here on the important tradeoff in the considered context between the cache insertion rate and misses of the second type, and study cache performance neglecting misses of the first type ("capacity misses"). Cache replacement policies and their impact on capacity misses have been well-studied in prior work. We note, however, that as the cache insertion policy becomes more selective, as we advocate here, it becomes more and more likely that content inserted into the cache will experience additional access(es). Cache replacement policies should then become more effective, since a higher fraction of the cache content will have experienced at least one further access since being inserted into the cache, providing more information to the cache replacement policy (for example, more useful reference count data for an LFU policy, or more meaningful LRU stack positions for an LRU policy).

We consider here content delivery applications in which content item popularities vary over time, and large numbers of new content items are continually being added. Our model of edge network caching performance is not, therefore, based on an assumption that there is some stationary popularity distribution for a fixed set of content items. Instead, we consider how many times a "new" content item that is requested, i.e. an item that has not been requested previously from the edge network, will be requested in total from that network. Based on our characterization results from Section 3.1, our model makes the assumption that this total number of requests follows a discrete power-law distribution with parameter $\alpha > 1$.

Under this assumption, and neglecting capacity misses, a cache using a *cache on first request* policy would experience a miss rate (probability) of

$$P(\text{cache miss}) = \frac{\sum_{i=1}^{\infty} i^{-\alpha}}{\sum_{i=1}^{\infty} i^{-\alpha} i} = \frac{\sum_{i=1}^{\infty} i^{-\alpha}}{\sum_{i=1}^{\infty} i^{-\alpha+1}}. \quad (2)$$

The cache insertion rate (fraction of requests for which the requested video is loaded into the cache) with this policy is identical to the miss rate.

4.2 Cache on kth Request

Scenarios in which there are many one-timers motivate consideration of more cautious cache insertion policies that require observation of multiple requests for a content item before inserting that item into the cache. Such a policy could be fruitful, however, only if the probability of additional requests for a content item, given that k requests have already been observed, is smaller for $k = 1$ than for larger k. Denoting the probability $P(\text{more requests} \mid k \text{ already observed})$ by $P(k^+|k)$, given our power-law distribution assumption we have

$$P(k^+|k) = \frac{\sum_{i=k+1}^{\infty} i^{-\alpha}}{\sum_{i=k}^{\infty} i^{-\alpha}}. \quad (3)$$

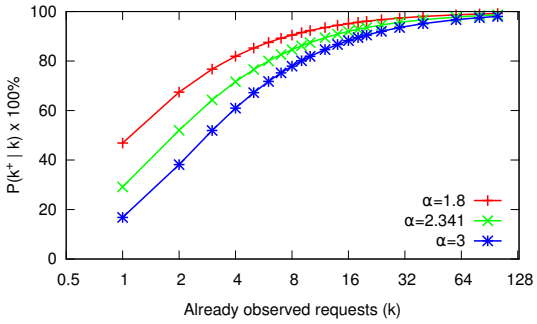


Fig. 8. $P(k^+|k)$ plotted as a percentage, as a function of k.

Figure 8 plots $P(k^+|k) \times 100\%$ for $\alpha = 2.341$, the value yielding the best fit for our trace data, as well as for somewhat smaller ($\alpha = 1.8$) and larger ($\alpha = 3.0$) shape parameter values. Note that $P(k^+|k)$ is an increasing function of k. Owing to the "decreasing failure rate" characteristic of a power-law distribution, the more requests that have been observed, the more likely that future requests will be observed.

Given our assumptions, a cache using a *cache on kth request* policy, for integer $k \geq 1$, would have cache miss and insertion rates given by

$$P(\text{cache miss}) = 1 - \frac{\sum_{i=k}^{\infty} i^{-\alpha}(i-k)}{\sum_{i=1}^{\infty} i^{-\alpha+1}}, \quad (4)$$

and

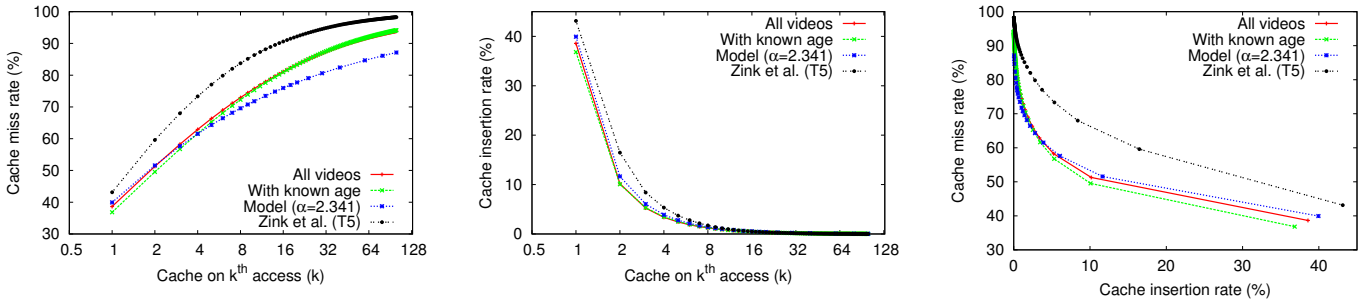
$$P(\text{cache insertion}) = \frac{\sum_{i=k}^{\infty} i^{-\alpha}}{\sum_{i=1}^{\infty} i^{-\alpha+1}}. \quad (5)$$

To understand the performance tradeoff using the *cache on kth request* policy, we use both trace-driven simulations and our analytic model. For the simulations, we use all requests seen at the campus network over the 20-month measurement period. To avoid transient effects, we use a warm-up period of 6 months and only calculate performance statistics over requests during the last 14 months of the trace. Figures 9(a) and 9(b) show the miss rate and insertion rate, respectively, as a function of k. We note that $k = 1$ corresponds to indiscriminate caching.

Although the model somewhat underestimates the cache miss rate (likely due to the model only approximating the distribution of total number of requests), as seen in Figure 9(c), the model still captures the general performance tradeoff between the cache miss and insertion rates.

For comparison, we also include curves for the longest public edge-network trace of YouTube accesses that we are aware of [37]. This trace is referred to as T5 by Zink et al. [37] and captures all YouTube views observed on their campus network over a two week period (Jan. 29, 2008 to Feb. 12, 2008). Due to the shorter trace duration, we did not employ any warmup period for this dataset (as we did for our much longer continuous trace). The trace includes 611,968 unique requests spread over 263,970 unique YouTube videos.[4] The somewhat higher miss rates observed for this dataset than for our dataset can partially be explained by the lack of

4. In the original reference it is stated that T5 includes 303,331 unique videos. The difference in the reported numbers is that we count requests of the form `01FCHBDmD7s&signature` and `01FCHBDmD7s&start` as being requests to the same video `01FCHBDmD7s`.



(a) Cache miss rate (b) Cache insertion rate (c) Tradeoff

Fig. 9. Performance tradeoffs using the *cache on k^{th} request* policy. Note that $k = 1$ corresponds to indiscriminate caching.

warmup period. In particular, some of the misses at the start of this trace would have been "hits" if we could have "warmed up" the cache. The shorter trace also causes the hit rates of the inserted videos to reduce (and miss rates to go up) since videos inserted into the cache towards the end of a trace in some cases do not see additional views due to the trace ending before additional views occur, rather than due to the videos not being viewed again.

While the absolute values differ somewhat from the values observed for our trace, the qualitative behavior is similar for both traces. This is perhaps not surprising, as both long-term and short-term viewing patterns at the server side have been shown to be similarly well-captured with various heavy-tailed distributions [15], [29]. We also note that due to the ephemeral nature of this content, for many videos, most of the views happen relatively soon after the first view of the video. In the remainder of the paper we will focus only on our (much longer) trace, for which we also have richer and complementing information both from the network trace and the YouTube service.

4.3 Lower Bound Performance

To glean some insight into the potential performance advantage that predictive algorithms could achieve, we consider the performance possible with policies that have knowledge of *future* requests. For this evaluation, we consider two *optimal oracle* policies.

For the first oracle policy, called *optimal oracle, perfect knowledge*, we assume that the oracle always knows *exactly how many* future requests a video will have. Clearly, given this knowledge, the optimal performance tradeoff between the cache miss and insertion rates is achieved by a threshold-based policy that always caches a video at the time of the first request whenever the video will have at least k requests, where k is a policy parameter chosen according to the desired miss/insertion rate tradeoff. Such a policy will have the same insertion rate as the *cache on k^{th} request* policy (equation (5)), but have a reduced cache miss rate:

$$P(\text{cache miss}) = 1 - \frac{\sum_{i=k}^{\infty} i^{-\alpha}(i-1)}{\sum_{i=1}^{\infty} i^{-\alpha+1}}, \tag{6}$$

due to the caching taking place already at the time of the first request.

For the second oracle policy, called *optimal oracle, binary knowledge*, we assume that the oracle always knows *if* a requested video will be requested again, but does not know how many future requests the video will have, in the cases where there are future requests. Given this knowledge, the best performance tradeoff is achieved by caching the video on the k^{th} request if there will be additional requests for that video. Clearly, this policy will have the same insertion rate as a *cache on $(k+1)^{st}$ request* policy, but will see one additional cache hit per insertion, compared to that policy. Hence, we can calculate the cache insertion rate and miss rate as follows:

$$P(\text{cache insertion}) = \frac{\sum_{i=k+1}^{\infty} i^{-\alpha}}{\sum_{i=1}^{\infty} i^{-\alpha+1}}, \tag{7}$$

$$P(\text{cache miss}) = 1 - \frac{\sum_{i=k+1}^{\infty} i^{-\alpha}(i-k)}{\sum_{i=1}^{\infty} i^{-\alpha+1}}. \tag{8}$$

We note that the first oracle policy provides an overall lower bound and the knowledge used by the second oracle policy is comparable to the knowledge used by Belady's algorithm [3], for example, as it only leverages information about the next request of the video. However, different from the goal with Belady's algorithm, we are concerned here with the tradeoff between the cache insertion rate and the cache miss rate. Clearly, a policy that only looks ahead to the next request, if any, is non-optimal here, as such a policy may suggest insertions of two-timers when only at least three-timers should be cached to achieve the desired tradeoff between insertion cost and miss cost.

4.4 Oracles with Limited Prediction

To understand the impact of the limited prediction capabilities facing real systems, we next consider oracles with perfect prediction capability for some videos, but only limited capability for others. In particular, we consider the two extremes in which the oracle is capable of predicting the exact number of future requests either (i) for all videos that will have at least X viewing requests, or (ii) for all videos that will have at most X viewing requests. We call these two oracle policies *limited oracle, perfect top-hitters* and *limited oracle, perfect ephemeral*, respectively.

Limited oracle, perfect top-hitters: First, assume that the oracle can predict exactly how many requests all videos that will have at least X requests will have in the future, but that it will not be able to make predictions for less popular videos. In this case, the optimal policy would always cache on the k^{th} request, where k is a policy parameter determining the tradeoff between miss and insertion rates, except

for popular videos (with $x_i > \max[k, X]$) which the oracle would cache on their *first* request.

To analyze the performance of such a policy, we first note that it has the same insertion rate as any other policy that caches those videos with at least k total requests, including the *cache on k^{th} request* policy (equation (5)). However, in contrast to the oracle with perfect knowledge of the future requests of all videos, this policy would see $(k-1)$ fewer hits for all videos with k to $X-1$ total requests; the resulting total miss rate is given by:

$$P(\text{cache miss}) = 1 - \frac{\sum_{i=k}^{\infty} i^{-\alpha}(i-1) - (k-1)\sum_{i=k}^{X-1} i^{-\alpha}}{\sum_{i=1}^{\infty} i^{-\alpha+1}}. \quad (9)$$

Figure 10 shows the performance gaps between the *cache on k^{th} request* policy, the two "perfect" oracle policies described in Section 4.3, as well as the *limited oracle, perfect top-hitters* policy with $X = 2$, 5, 10, and 20. Results are shown for both the analytic model (Figure 10(a)) and trace-driven simulations (Figure 10(b)).

The analytic and simulation results for the most part look qualitatively similar, although the simulations (due to the 6 months warm-up period) see a smaller relative insertion rate for the oracle policies (e.g., as the oracle policies have already inserted many videos during the warm-up period).

Comparing the *cache on k^{th} request* policy and *optimal oracle, perfect knowledge*, we see that there are significant advantages to having knowledge of future requests. Perhaps most interesting is the observation that the gap is the biggest for small k, thus suggesting that the biggest improvements are due to knowing which videos should *not* be cached, rather than on predicting the very popular videos that are likely to be cached eventually anyway, using a simple *cache on k^{th} request* policy, for example.

Looking closer at the penalty of the oracle with limited prediction, that can only predict the number of future requests to popular videos, we see that X must be small before *limited oracle, top-hitters* yields substantially better performance than the basic *cache on k^{th} request* policy. For example, if the oracle can only tell us which videos will see more than 10-20 requests, the oracle does not help much, compared to the simple *cache on k^{th} request* policy.

Limited oracle, perfect ephemeral: At the other end of the spectrum is an oracle that can perfectly predict the exact number of requests for the videos with at most X requests, but that can only predict that other videos will have more than X requests, not how many requests beyond that. As for the other oracle policies with knowledge about the number of future requests, the optimal policy for such an oracle is to always cache at first request whenever it knows there will be more than k requests, where k is a policy parameter, and otherwise not cache until (possibly) reaching k requests. As this later caching can only happen when $X < k$, the cache miss rate of this policy is the same as for *optimal oracle, perfect knowledge* whenever $k \leq X$ and equal to that of the *cache on k^{th} request* policy otherwise. For a given k, the cache insertion rate is the same for all these three policies.

Figure 11 shows results for $X = 2$, 5, 10, and 20. As per the description of this oracle, we note that its performance follows that of the overall optimal for values of k up to the threshold X, after which point it starts tracking the *cache at k^{th} request* policy. Comparing with the results for *limited oracle, perfect top-hitters* (Figure 10), we note that there is a significant advantage to being able to predict the number of requests for the least popular videos, compared to being able to accurately predict the requests for popular videos.

Overall, these results confirm that there are greater benefits to being able to accurately discriminate among videos that will receive few requests, for example predicting a one-timer versus a 5-timer, compared to predicting the number of future requests for popular videos. Unfortunately, predicting exactly how many future requests unpopular videos will have on an edge network is non-trivial.

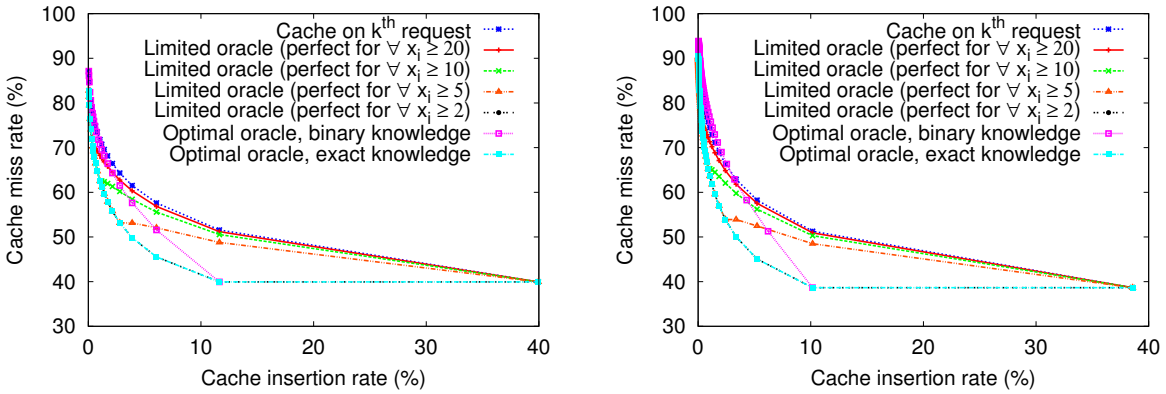
4.5 Read/write Penalty Analysis

We next take a closer look at the tradeoff between the read/write ratio (potentially important in flash-based SSD systems) and the cache miss rate. Figure 12 summarizes the performance tradeoff between cache misses and the read/write ratio for alternative policies. Regardless of policy, we note that improved read/write ratio comes at the cost of higher cache miss rate. Again, there are clear advantages to oracles that can predict the number of requests to videos with few requests. For example, even limited ($\forall x_i \leq 2$) or limited ($\forall x_i \leq 5$) can achieve a read/write ratio of 4 or 16, respectively, while achieving the same cache miss rate as the oracle with perfect knowledge of all future requests.

To glean some insight into the desirable value of the k parameter for an SSD-based caching system, we use a back-of-the-envelope example scenario and calculate the expected read/write ratio and the length of the time span over which the write volume would equal the SSD capacity for two example SSD sizes. For simplicity, we assume that the campus load (with a total of 5.5M requests) is evenly spread across the 20 months, resulting in an average of 275K requests per month. Furthermore, we consider a small SSD system with room for 10,000 videos and a larger system with room for 50,000 videos. Using Little's law together with the read/write ratio and the cache insertion rates for different k, we can now estimate the length of the time span over which the write volume would equal the SSD capacity. Figure 13 summarizes the results. Note that for a larger network with more users and higher rate of requests per month, the time spans shown would correspondingly decrease.

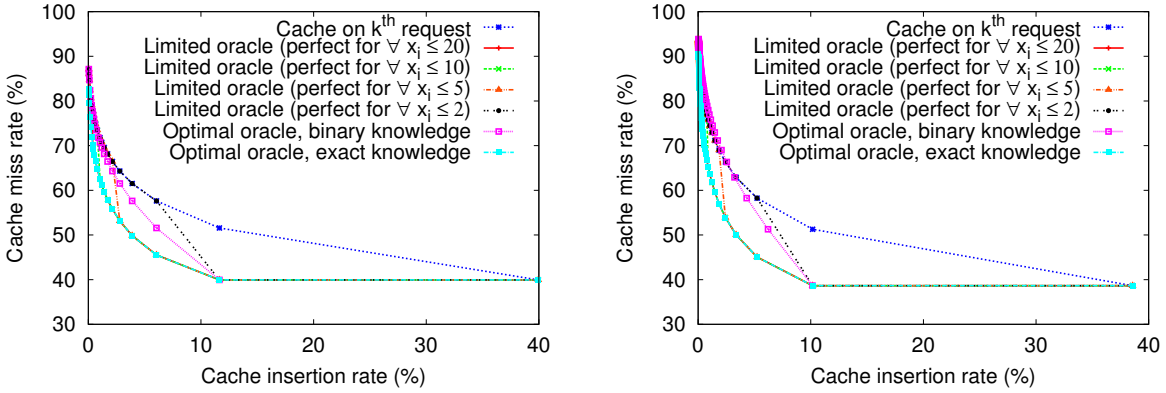
5 Basic Policies with Imperfect Prediction

Unfortunately, no edge network has access to an accurate oracle. We next take a closer look at the performance gains of two basic policies that leverage observed biases in request frequencies to improve upon the *cache on k^{th} request* policy. For this purpose, we identify videos that are more (or less) likely to see future requests than a random video that sees at least k requests. More specifically, in the following we explore both policies that adapt when to cache on k^{th} access, based on the time between accesses, such as to avoid caching videos that are less likely to see future requests, and policies that more aggressively cache videos that are less likely to be one-timers (based on their age at the time of first access).



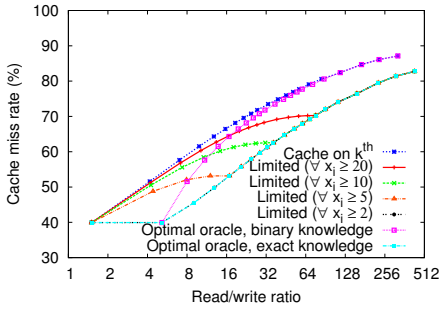
(a) Model with $\alpha = 2.341$ (b) Trace-driven simulation

Fig. 10. Performance with *limited oracle, perfect top-hitters*, compared to the *cache on k^{th} request* policy, and the two lower-bound oracle policies with "perfect" and "binary" (future) knowledge, respectively.

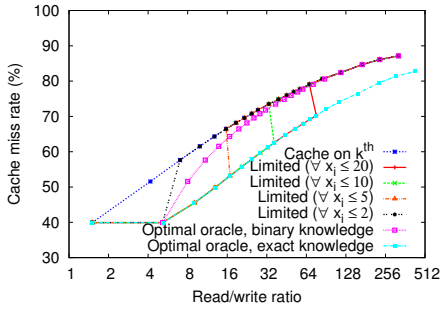


(a) Model with $\alpha = 2.341$ (b) Trace-driven simulation

Fig. 11. Performance with *limited oracle, perfect ephemeral*, compared to the *cache on k^{th} request* policy, and the two lower-bound oracle policies with "perfect" and "binary" (future) knowledge, respectively.



(a) Top-hitter predictor



(b) One-timer predictor

Fig. 12. The performance tradeoff between cache misses and the read/write ratio of the *cache on k^{th} request* policy relative to oracles with different capabilities. (Model with $\alpha = 2.341$)

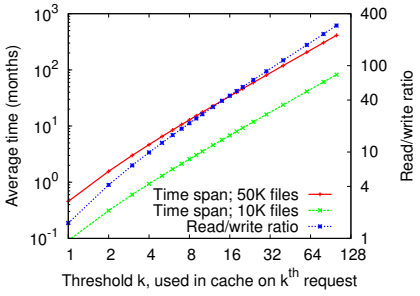


Fig. 13. The length of the time span over which the write volume would equal the SSD capacity, assuming an SSD that could hold 10,000 or 50,000 videos, respectively, as a function of k.

5.1 Inter-request Threshold Cache on k^{th} Request

We have found that most often a video's second request comes soon after the first request (Figure 14), and that a video that has a short time between its first two requests on average sees a higher number of future requests (Figure 15) and is more likely to be requested again (Figure 16).

Motivated by these observations, we define a threshold-based policy called *inter-request threshold cache on k^{th} request*. This policy uses the time between the $(k-1)^{st}$ and k^{th} requests to determine if the video should be cached at the second of these requests or not. Using a time threshold T, the policy only caches on the second of these requests if the preceding request took place less than T ago. This policy reduces the fraction of videos that are cached, while prioritizing videos that are likely to see relatively more requests.

5.2 Age Threshold Cache on 1^{st} Request

For $k = 2$, the *inter-request threshold cache on k^{th} request* policy is a more conservative caching policy than the baseline *cache on 2^{nd} request* policy. A second possible way to improve on *cache on 2^{nd} request*, would be to be more aggressive

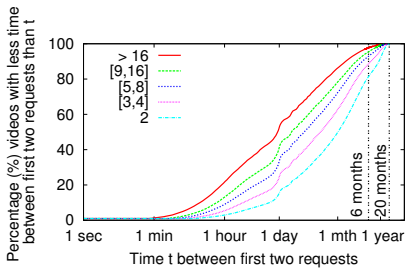


Fig. 14. Distribution of time between first and second requests, for videos with different numbers of requests (legend).

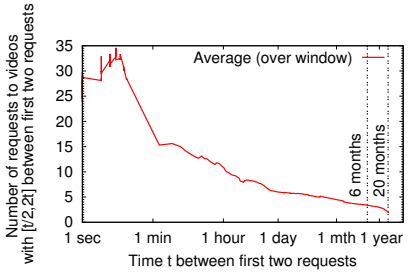


Fig. 15. The expected number of requests $E[x_i|t]$ for videos whose first inter-request time is at least $t/2$ and at most $2t$, plotted as a function of the logarithmic mid-point t.

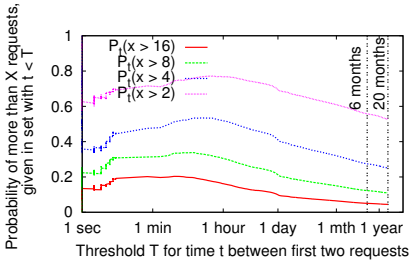


Fig. 16. Probability $P(x_i > X|t < T)$ of more than X requests, conditioned on the first inter-request time t being less than T.

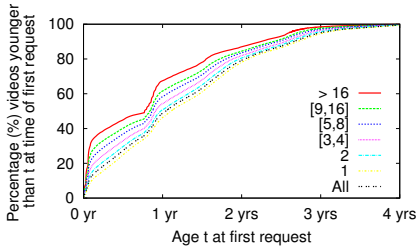


Fig. 17. Video age distribution at time of first viewing request, for videos with different number of requests (legend).

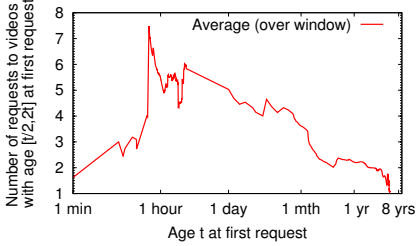


Fig. 18. The expected number of requests $E[x_i|t]$ for videos with age between $t/2$ and $2t$ when first requested, plotted as a function of the logarithmic mid-point t.

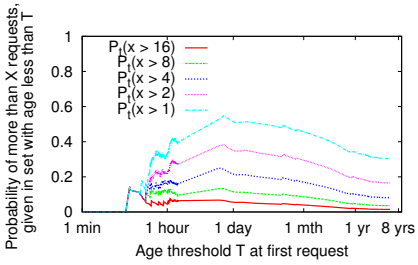


Fig. 19. Probability $P(x_i > X|t < T)$ of more than X requests, conditioned on the age t at the time of the first request being less than T.

with caching of videos that are more likely to see additional views. As indicated by the one-timer biases identified in Section 3, one approach may be to cache videos of certain video categories (that see more views per video, on average, for example) on their first request. More advanced predictors could include statistics about the past success of the video uploader, the video's current global popularity, and other video specific metrics. While we have considered such policies, none have provided substantial improvements beyond just randomly picking a subset of videos to cache on their first request. The main reason for this is due to the relatively small differences among the categories in the probability of a video being a one-timer. For example, referring to Figure 6, we note that the fraction of one-timers typically differs by less than a factor of two between the video categories. This only allows for small biases. Rather than trying to combine categorizations and metrics to determine the best possible such predictor (possibly using some meta data not available to us), we instead define only a basic threshold policy, *age threshold cache on 1^{st} request*, which uses the video age to determine whether to cache a video when first requested.

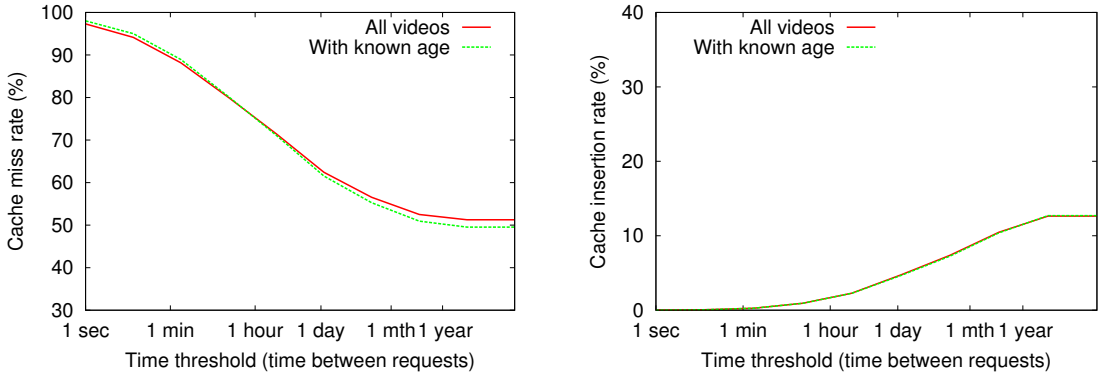
The use of the video age at the time of the first request is motivated by big differences in the expected number of future views to videos of different ages at the time of the first request (Figure 18), and by the observation that videos younger than some threshold age T at the time of their first request would be more likely to see additional views (Figure 19) than older videos. For example, as seen in Figure 19, the average number of requests for videos with age value (at the time of the first request) of t less than 1 month is typically at least twice that for videos with age value t of a year or more. As the videos that are less than 1 month old at the time of the first request correspond to a fairly small fraction of the total number of videos (e.g., roughly 5-25% according to Figure 17), adding such a video to the cache at the time of its first request may provide an attractive tradeoff.

5.3 Performance Comparison

Figures 20 and 21 show the basic performance tradeoff between cache misses and cache insertion rate, as a function of the thresholds used in the two threshold-based policies. Figure 22 summarizes their performance tradeoff relative to the *cache on k^{th} request* policy, and also shows results for a hybrid policy that uses both threshold policies simultaneously. While both policies are able to provide some minor improvements, neither is even close to achieving the performance tradeoffs seen by the oracle policies. The lack of bigger gains is simply an effect of prediction of one-timers being a very difficult problem. Note that both threshold policies may be relatively complex to parameterize in practice, particularly if parameterization was dependent on other factors such as video category. This complexity is likely not justified by the limited performance gains they provide.

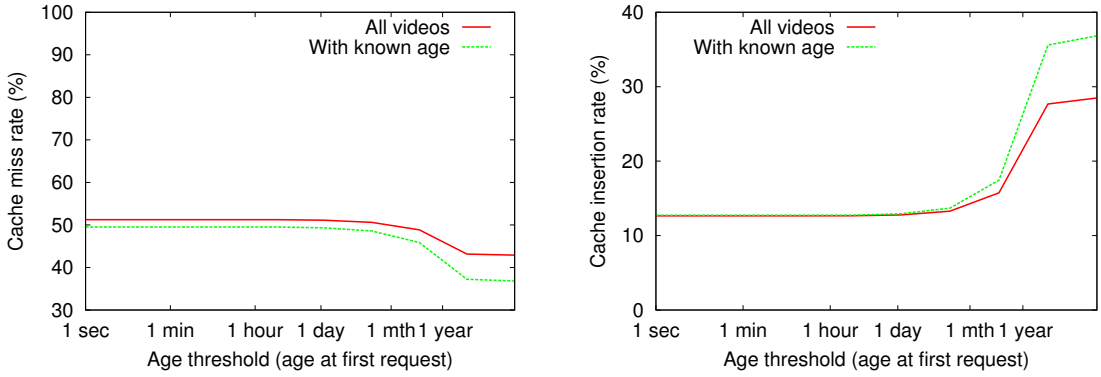
The results show that the thresholds cannot be naively picked, and that it is better to use a separate policy for each region of the tradeoff curve. For example, the age-based threshold policy is good when wanting to cache on first request, but has a much lower probability that an item added to the cache will be requested again compared to the *cache on k^{th} request* policy when $k \geq 2$. In general, the bias in the success probabilities of any subset of videos should be weighted against the random success probability of a video being requested again, as captured by equation (3), for example.

To put the observed gains in perspective, we next consider the impact of imperfect, but still good prediction,



(a) Cache miss rate (b) Cache insertion rate

Fig. 20. Performance tradeoffs using the *inter-request threshold cache on k^{th} request* policy for $k = 2$.



(a) Cache miss rate (b) Cache insertion rate

Fig. 21. Performance tradeoffs using the *age threshold cache on 1^{st} request* policy.

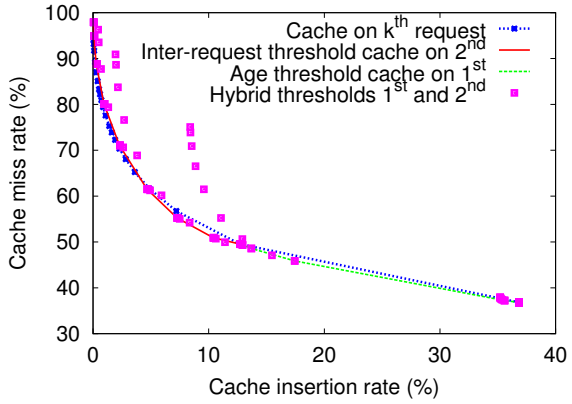


Fig. 22. Summary of the cache performance tradeoffs of the two threshold-based policies, for all videos with known age.

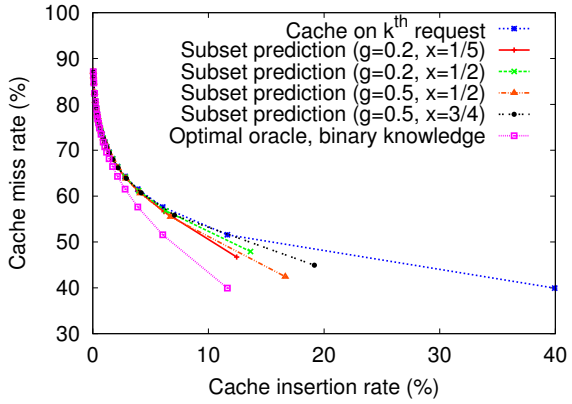


Fig. 23. Performance when caching biased subset of the videos, but the prediction is not perfect.

within our modelling framework. For simplicity, assume that we are able to identify a subset of the videos for which we are able to predict the fraction of videos that have a higher than random likelihood that they will be requested again after the k^{th} request. In particular, assume that a fraction g of the videos with at least k requests belong to this subset, and that the proportion of these for which no more requests will be observed is reduced by a factor $x \leq 1$ compared to randomly selected videos. Of course, for such prediction to be possible we must have:

$$\frac{1 - x\frac{n}{N}}{1 - \frac{n}{N}}(1 - P(k^+|k)) \leq 1, \tag{10}$$

where $P(k^+|k)$ is given by equation (3). Otherwise, the probability for the other videos (not in the identified subset) to see more requests would be negative.

Consider now the performance of a policy in which all videos in the identified subset are cached on the k^{th} access, and all other videos on the $(k+1)^{st}$ access. The cache insertion rate with this policy is given by the cache insertion rate with the *cache on (k+1)st request* policy, plus the rate of inserting items from the identified subset at their k^{th} request that will not be requested again. Hence, the total insertion

rate can be calculated as:

$$P(\text{cache insertion}) = \frac{\sum_{i=k+1}^{\infty} i^{-\alpha} + \sum_{i=k}^{\infty} i^{-\alpha} g x (1 - P(k^+|k))}{\sum_{i=1}^{\infty} i^{-\alpha+1}}, \quad (11)$$

where $P(k^+|k)$ is given by equation (3). Furthermore, such a policy will always see the same cache hits as a *cache at* $(k+1)^{st}$ *request* policy, plus the cache hits due to the videos added early on the k^{th} request. The cache miss rate can hence be calculated as:

$$P(\text{cache miss}) = 1 - \frac{\sum_{i=k+1}^{\infty} i^{-\alpha}(i-k-1) + \sum_{i=k}^{\infty} i^{-\alpha} g (1 - x(1 - P(k^+|k)))}{\sum_{i=1}^{\infty} i^{-\alpha+1}} \quad (12)$$

Figure 23 shows the performance tradeoffs seen by such a policy for relatively optimistic prediction accuracy. In this example, we show curves for the case in which the identified set encompasses either 20% or 50% of the videos and the videos in this set have a probability of not being re-requested that is 1/5, 1/2, or 3/4 of the average. Of course, as shown in Figure 8, these rates are relatively small when k grows big. Therefore the biggest improvements are when k is small and the insertion rate is relatively high.

However, despite relatively optimistic prediction accuracy, the improvements are moderate compared to the lower bound policy. In fact, for the case with $g = 1/2$ and $x = 3/4$, for which the relative probabilities (in subset versus not in subset) of not getting additional requests differ by a factor 5/3 (almost 2), there are only marginal improvements compared to a policy that picks random videos to cache (which would result in a point on the curve for the *cache at* k^{th} *request* policy).

These results show that high prediction accuracy is needed for the videos with few references for there to be substantial improvements. However, as seen here and from our other results, basic one-dimensional categorization and threshold policies are only able to provide very moderate/limited improvements.

6 Related Work

There is considerable current interest in caching-related research. Much of this interest is motivated by the central role caching has in both existing and emerging content distribution systems. Even small improvements in caching strategies can have the potential to result in significant cost savings and improvements in perceived user experience.

Today, content delivery is responsible for the majority of the Internet bandwidth usage. Carefully designed Content Delivery Networks (CDNs) are used to effectively serve large populations of geographically distributed users [16], [24], and operators are partnering directly with major content providers such as Google (e.g., YouTube videos) and Netflix to help reduce their access bandwidth usage, with the help of content provider driven caches within the operators' networks. Within the content delivery context, researchers have recently explored the best approaches to reduce the delivery costs through careful cache placement [25], networks of caches [19], cooperative cache management [7], and cache replica selection [9], for example. Within the context of Information Centric Networks (ICNs), it has been argued that opportunistic edge-based caching only near the end-users outperforms on-path caching approaches as traditionally assumed in most ICN architectures [13], further motivating scenarios such as that considered in this paper.

During the 90's a large number of papers studying cache replacement policies were published [2], [34]. These include the analysis of basic replacement policies such as Least Recently Used (LRU) and First-in-First-out (FIFO) [14], as well as more advanced replacement policies such as Greedy Dual-Size with Frequency (GDSF) [1]. Recently, the Che approximation [11] has been used to evaluate and relate various cache replacement strategies under a rich set of scenarios [4], [20], [28]. Rather than studying the cache replacement policies, the focus in this paper is on which contents to cache in the first place, not which contents to remove.

Most existing cache modelling literature (both old and new) assumes stationary content popularities and the independent reference model. Already in 1999, Wolman et al. [36] challenged the current models, showed that capacity misses are rare, and suggested that misses instead typically are due to continual generation of new or modified content. Our data supports the insights by Wolman et al. [36]. Using our trace data, we identify a power-law relationship that we use to model and estimate cache miss rates and cache insertion rates under different policies.

Traditionally, cache insertion and prefetching decisions typically leverage spatial and temporal request locality, or hints from the servers [30], [31]. However, increasingly, request biases are discovered using careful data mining [5], [33]. Rather than designing new prediction algorithms, we take a closer look at the best possible performance improvements such algorithms potentially could enable.

Other researchers have characterized the YouTube popularity dynamics, both globally [5], [6], [8], [10], [18], [22] and at the edge [21], [37]. Perhaps, most closely related are the works by Gill et al. [21] and Zink et al. [37], in which the authors characterize the YouTube traffic on their respective campus networks. While both studies are much shorter in duration and neither study looks closely at the one-timers, both studies observe that there is a substantial fraction of one-timers, one of the observations that originally motivated our work.

Recently, Maggs and Sitaraman [26] reported a similarly large footprint of one-timers (called "one-hit-wonders" in their paper) among the web objects and videos served by one of Akamai's server clusters, with roughly 75% of the objects only being accessed once. Motivated by this observation, they, similar to us, investigate the effectiveness of a cache-on-second-hit rule, equal to the special case of our *cache on* k^{th} *request* policy with $k = 2$. Their production results demonstrate that such a rule can help increase byte hit rates, decrease disk writes, and as an effect also reduce disk latencies. Our work complements the work of Maggs and Sitaraman by presenting characterization and modeling results for understanding the effect of one-timers and other videos that receive few views ("k-timers" for small k), and by considering alternative policy variations

for when to cache. In addition to providing these aspects, we also present lower bounds, allowing us to delimit the best possible cache tradeoffs, and provide insights on the possible room for further improvements.

Heavy-tailed popularity distributions, such as the directly related Zipf distribution (rank frequency relationship) and the power-law distribution (frequency of videos with different numbers of views) have been widely observed, including in the two edge-based characterization studies mentioned above [21], [37]. Other researchers have distinguished between short-term and long-term popularity, and studied popularity dynamics [6], [29]. In this work, in contrast to most prior work, we pay particular attention to the one-timers and the least popular content items. We provide the first characterization and modeling study focused on the impact on caching of these content items, and evaluate the value of predicting which content items are most likely to only be requested one or only a handful of times. In contrast, most prior work has focused on characterizing [5], [10], [21] or predicting [17], [33], [35] the popular contents. For contexts in which there is considerable ephemeral content, we show that a basic *cache on k^{th} request* policy (as motivated by equation (3), for example) provides most of the benefits that are likely to be achievable in practice.

7 Conclusions

This paper makes three contributions. First, we collected and analyzed a longitudinal dataset of all YouTube video accesses from users on an edge network over a 20-month period. Our study found that most accessed videos received few views (e.g., 71% of the requested videos were only requested once within the measurement period) and that the requests per accessed video can be accurately modelled using a power-law distribution with scale parameter $\alpha = 2.34$. A similar fractions of one-timers (75%) was recently observed by Maggs and Sitaraman [26] for an Akamai workload. Interesting future work could consider alternative video services such as Hulu, for example, or other types of content.

Second, we use both a novel workload model for ephemeral content and trace-driven simulations to study the performance of alternative edge caching policies, including indiscriminate caching and *cache on k^{th} request* for different k. The latter policies are found able to greatly reduce the fraction of the accessed content items that are inserted into the cache, at the cost of relatively modest increases in cache miss rate.

Finally, we assess the potential room for improvement in these policies through use of content characteristics such as the time between requests and the content age at first access to predict the likelihood of future requests. Although we find that there is room for substantial improvement when comparing *cache on k^{th} request* performance to that of a perfect "oracle" policy, achieving such improvements would require the prediction of the number of future requests to the content items that are the least popular. Unfortunately, this problem is both difficult and not well explored, as most content popularity prediction research has focused on predicting the most popular contents.

Acknowledgements

The edge-network collection was done while the first author was a research associate at the University of Calgary. We thank Carey Williamson and Martin Arlitt for providing access to this dataset. This work was supported by funding from Center for Industrial Information Technology (CENIIT), and the Natural Sciences and Engineering Research Council (NSERC) of Canada.

References

[1] M. Arlitt, L. Cherkasova, J. Dilley, R. Friedrich, and T. Jin. Evaluating content management techniques for web proxy caches. In *Proc. ACM SIGMETRICS*, 2000.
[2] G. Barish and K. Obraczke. World wide web caching: trends and techniques. *IEEE Communications Magazine*, 38(5):178–184, May 2000.
[3] L. A. Belady. A study of replacement algorithms for a virtual-storage computer. *IBM Systems Journal*, 5:78–101, 1966.
[4] D. S. Berger, P. Gland, S. Singla, and F. Ciucu. Exact analysis of TTL cache networks. *Performance Evaluation*, 2014.
[5] Y. Borghol, S. Ardon, N. Carlsson, D. Eager, and A. Mahanti. The untold story of the clones: Content-agnostic factors that impact YouTube video popularity. In *Proc. ACM SIGKDD*, 2012.
[6] Y. Borghol, S. Mitra, S. Ardon, N. Carlsson, D. Eager, and A. Mahanti. Characterizing and modeling popularity of user-generated videos. In *Proc. IFIP PERFORMANCE*, 2011.
[7] S. Borst, V. Gupta, and A. Walid. Distributed caching algorithms for content distribution networks. In *Proc. IEEE INFOCOM*, 2010.
[8] A. Brodersen, S. Scellato, and M. Wattenhofer. YouTube around the world: Geographic popularity of videos. In *Proc. WWW*, 2012.
[9] N. Carlsson, D. Eager, A. Gopinathan, and Z. Li. Caching and optimized request routing in cloud-based content delivery systems. *Performance Evaluation*, 79:38–55, Sept 2014.
[10] M. Cha, H. Kwak, P. Rodriguez, Y. Ahn, and S. Moon. I tube, you tube, everybody tubes: Analyzing the world's largest user generated content video system. In *Proc. ACM IMC*, Oct. 2007.
[11] H. Che, Y. Tung, and Z. Wang. Hierarchical web caching systems: Modeling, design and experimental results. *IEEE J. Sel. Areas Commun.*, 20(7):1305–1314, 2002.
[12] A. Clauset, C. Shalizi, and M. Newman. Power-law distributions in empirical data. *SIAM Review*, 51(4):661–703, Nov. 2009.
[13] A. Dabirmoghaddam, M. Mirzazad-Barijough, and J. J. Garcia-Luna-Aceves. Understanding optimal caching and opportunistic caching at "the edge" of information-centric networks. In *Proc. ACM ICN*, 2014.
[14] A. Dan and D. Towsley. An approximate analysis of the LRU and FIFO buffer replacement schemes. In *Proc. ACM SIGMETRICS*, 1990.
[15] G. Dan and N. Carlsson. Power-law revisited: A large scale measurement study of P2P content popularity. In *Proc. IPTPS*, 2010.
[16] J. Dilley, B. Maggs, J. Parikh, H. Prokop, R. Sitaraman, and B. Weihl. Globally distributed content delivery. *IEEE Internet Computing*, 6(5), Sept/Oct. 2002.
[17] F. Figueiredo. On the prediction of popularity of trends and hits for user generated videos. In *Proc. WSDM*, pages 741–746, 2013.
[18] F. Figueiredo, F. Benevenuto, and J. M. Almeida. The tube over time: Characterizing popularity growth of YouTube videos. In *Proc. ACM WSDM*, 2011.
[19] N. C. Fofack, P. Nain, G. Neglia, and D. Towsely. Analysis of TTL-based cache networks. In *Proc. VALUETOOLS*, 2012.
[20] C. Fricker, P. Robert, and J. Roberts. A versatile and accurate approximation for LRU cache performance. In *Proc. ITC*, 2012.
[21] P. Gill, M. Arlitt, Z. Li, and A. Mahanti. YouTube traffic characterization: A view from the edge. In *Proc. IMC*, Oct. 2007.
[22] G. Gursun, M. Crovella, and I. Matta. Describing and forecasting video access patterns. In *Proc. IEEE INFOCOM*, 2011.
[23] M. A. Islam, D. Eager, N. Carlsson, and A. Mahanti. Revisiting popularity characterization and modeling for user-generated videos. In *Proc. IEEE MASCOTS*, San Francisco, CA, Aug. 2013.
[24] W. Jiang, S. Ioannidis, L. Massoulie, and F. Picconi. Orchestrating massively distributed CDNs. In *Proc. ACM CoNEXT*, 2012.
[25] P. Krishnan, D. Raz, and Y. Shavitt. The cache location problem. *IEEE/ACM Transactions on Networking*, 8(5):568–582, Oct. 2000.

[26] B. M. Maggs and R. K. Sitaraman. Algorithmic nuggets in content delivery. *ACM SIGCOMM Comput. Commun. Rev.*, 45(3):52–66, July 2015.
[27] A. Mahanti, N. Carlsson, A. Mahanti, M. Arlitt, and C. Williamson. A tale of the tails: Power-laws in Internet measurements. *IEEE Network*, 27(1):59–64, Jan/Feb. 2013.
[28] V. Martina, M. Garetto, and E. Leonardi. A unified approach to the performance analysis of caching systems. In *Proc. IEEE INFOCOM*, 2014.
[29] S. Mitra, M. Agrawal, A. Yadav, N. Carlsson, D. Eager, and A. Mahanti. Characterizing web-based video sharing workloads. *ACM Trans. on the Web*, 5(2):8:1–8:27, May 2011.
[30] J. C. Mogul. Hinted caching in the web. In *Proc. ACM SIGOPS European Workshop: Systems Support for Worldwide Applications*, 1996.
[31] V. N. Padmanabhan and J. C. Mogul. Using predictive prefetching to improve world wide web latency. *ACM SIGCOMM Computer Communication Review*, 26(3):22–36, 1996.
[32] V. Paxson. Bro: A system for detecting network intruders in real-time. In *Proc. USENIX Security Symposium*, 1998.
[33] H. Pinto, J. M. Almeida, and M. Goncalves. Using early view patterns to predict the popularity of YouTube videos. In *Proc. ACM WSDM*, 2013.
[34] S. Podlipnig and L. Böszörmenyi. A survey of web cache replacement strategies. *ACM Comput. Surv.*, 35(4):374–398, Dec. 2003.
[35] G. Szabo and B. A. Huberman. Predicting the popularity of online content. *Communications of the ACM*, 53(8):80–88, Aug. 2010.
[36] A. Wolman, G. Voelker, N. Sharma, N. Cardwell, A. Karlin, and H. Levy. On the scale and performance of cooperative web proxy caching. In *Proc. ACM SOSP*, 1999.
[37] M. Zink, K. Suh, Y. Gu, and J. Kurose. Characteristics of YouTube network traffic at a campus network - measurements, models, and implications. *Comput. Netw.*, 53(4):501–514, 2009.

BIOGRAPHIES

Dr. Niklas Carlsson is an Associate Professor at Linköping University, Sweden. He received his M.Sc. degree in Engineering Physics from Umeå University, Sweden, and his Ph.D. in Computer Science from the University of Saskatchewan, Canada. He has previously worked as a Postdoctoral Fellow at the University of Saskatchewan, Canada, and as a Research Associate at the University of Calgary, Canada. His research interests are in the areas of design, modeling, characterization, and performance evaluation of distributed systems and networks.

Dr. Derek Eager is a Professor in the Department of Computer Science at the University of Saskatchewan, Canada. He received the B.Sc. degree in Computer Science from the University of Regina, Canada, and the M.Sc. and Ph.D. degrees in Computer Science from the University of Toronto, Canada. His research interests are in the areas of performance evaluation, content distribution, distributed systems and networks.