Public Review for

Temporal Locality in Today's Content Caching: Why it Matters & How to Model It

Stefano Traverso, Mohamed Ahmed, Michele Garetto, Paolo Giaccone, Emilio Leonardi, and Saverio Niccolini

"I'm not attracted and we don't get along, but the timing is right."

New Yorker Cartoon by Barbara Smaller

If timing, as we are often reminded, is everything, it's also a critical part of today's caching system. The main reason is that the popularity of an item -- like a video on YouTube, as considered by the authors of this paper -- evolves quickly and it can easily fool a simplistic cache management strategy. In particular a heuristic model called *independent reference* is often used in evaluating caching networks even today, and this precisely states that popularity is fixed with time and that a new request is independent of the past. In reality requests typically occur in bursts, a phenomenon sometimes called temporal locality. How much of this locality matters in today's caching? Which methods can be used in practice to extract and recognize it when it occurs? Are there models that retain some of the simplicity of the independent reference model while accounting for temporal locality?

This paper answers the above questions by first conducting a longitudinal study of web-traffic inside a few operational Points of Presence (PoP) accounting for 60k users. It shows that temporal locality is present and that it generally implies that the performance derived for a simple caching scheme is unnecessarily pessimistic. It then makes two novel contributions: First, it proposes and evaluates a concrete method based on slicing and reshuffling to modulate temporal locality in a trace, showing how it directly impacts performance. Second, it builds a new temporal model of request generation that operates at a middle ground between mathematical simplicity and realism. The model reuses the above slicing and is inspired from Shot Noise Processes, a relatively mature tool of applied probability which allows one to sum independent processes that still exhibit themselves some temporal evolution. None of these tools and methods have ever been applied to caching for on-demand video or other content delivery.

All reviewers appreciated the substance and the *timing* of this work to propose a different model of cache request. In spite of various classical results, we still rely too blindly on the independent reference model, owing to its simplicity. In addition, this work was quoted as a good example of applying sophisticated analysis to a real data set. Reviews also highlighted some important limitations of these current results: Relying on a single cache algorithm (LRU) for all claims makes the paper less relevant in practice and it reduces its surprise factor since LRU has well known deficiencies. This is exacerbated by the fact that only relatively small caches are considered, with associated low hit probability. It is a fact that the authors recognize and is dictated by the limitations of the traces themselves. Finally, one of the main advantages of this new model (over cascade-based requests etc.) is to permit analysis, but that point is not *stricto sensu* demonstrated here. For those interested, the authors cite a follow up paper, available but not currently published, that focuses on this point.We hope that our community will react to this work with a renewed interest for a more accurate model of caching requests, and that this would eventually narrow the gap between practical design and mathematical analysis for future networks.

Public review written by
Augustin Chaintreau
Columbia University, USA

Temporal Locality in Today's Content Caching: Why it Matters and How to Model it

Stefano Traverso
Politecnico di Torino
Torino, Italy
stefano.traverso@polito.it

Mohamed Ahmed
NEC Labs Europe
Heidelberg, Germany
mohamed.ahmed@neclab.eu

Michele Garetto
Università di Torino
Torino, Italy
michele.garetto@unito.it

Paolo Giaccone
Politecnico di Torino
Torino, Italy
paolo.giaccone@polito.it

Emilio Leonardi
Politecnico di Torino
Torino, Italy
emilio.leonardi@polito.it

Saverio Niccolini
NEC Labs Europe
Heidelberg, Germany
saverio.niccolini@neclab.eu

ABSTRACT

The dimensioning of caching systems represents a difficult task in the design of infrastructures for content distribution in the current Internet. This paper addresses the problem of defining a realistic arrival process for the content requests generated by users, due its critical importance for both analytical and simulative evaluations of the performance of caching systems. First, with the aid of YouTube traces collected inside operational residential networks, we identify the characteristics of real traffic that need to be considered or can be safely neglected in order to accurately predict the performance of a cache. Second, we propose a new parsimonious traffic model, named the Shot Noise Model (SNM), that enables users to natively capture the dynamics of content popularity, whilst still being sufficiently simple to be employed effectively for both analytical and scalable simulative studies of caching systems. Finally, our results show that the SNM presents a much better solution to account for the temporal locality observed in real traffic compared to existing approaches.

Categories and Subject Descriptors C.2.1 [Network Architecture and Design]
General Terms: Theory, Measurement and Modelling
Keywords: Caching

1. INTRODUCTION

Content caching plays a critical role when operating networks by reducing the distance packets travel in the network. This results in lowered costs for operators and improved quality of service for end users. It is therefore no surprise that content caching plays a central role in many recent approaches aimed at improving network performance and utilisation, including; Information Centric Networking (ICN) [1, 2], resource management in cellular networks [3, 4], energy efficient networking [5], massively geographically distributed CDNs [6] and ISP-assisted CDNs [7].

However, when evaluating the benefits of their proposals, researchers are faced with the familiar question of what to evaluate their methodology on. This is particularly poignant for networks researchers looking to understand the impact of millions of users accessing many millions of content objects on a network. The usual methodology here is to perform trace-driven analysis, collecting traces of the phenomena in question and using these to drive simulations and emulations [3, 4, 6, 7, 8].

There are however two issues here. First, trace-driven analysis is effective only when large data sets are available, which, for most researchers is not always the case. Too often we are limited by the size and/or the availability of data sets, their diversity, or legal and privacy concerns. Second, this kind of analysis does not allow users to test potential changes in the traffic profile, such as changes in the popularity profiles of contents before they actually take place. In short, in order to evaluate the performance of caching systems, we must build models that enable us to evaluate effects that are too large to test in the wild, or for which there are no data traces available, or that correspond to scenarios which do not yet exist in operational systems.

In this work we focus on the problem of defining a model to accurately approximate content request rates, in particular for Video-on-Demand (VoD) systems. This problem is at the heart of determining the accuracy of the many previously published simulation and model-based results that aim to understand the performance of caching systems. This is important because given the preponderance of multimedia traffic in today's networks [8], it is reasonable to assume that the performance of the network as a whole is largely dominated by the effects of content delivery, and will become more so in the future.

Our contribution in this space is two-fold. First, with the aid of real traffic traces, we present the first quantification of the cache system performance errors introduced by the classical *Independent Reference Model* (IRM) [9] (the de-facto standard synthetic traffic model [10, 11]) in the context of on-demand video delivery. We show that the IRM leads to considerable errors, rendering its use in this context misguided at best. We believe this to be an important result, because, due its tractability, the IRM is still the standard assumption made in many works [5, 6, 12, 13]. As our second contribution, we propose a novel replacement to the IRM called the *Shot-Noise Model* (SNM). The SNM overcomes the IRM's limitations by explicitly accounting for the temporal locality in requests for contents, while still maintaining its desirable properties of simplicity and scalability. Finally, we validate the SNM's accuracy using real traffic traces.

The temporal locality in content requests, and its potential effects on caching systems have been previously studied by a number of works, including [14, 15, 16, 17]. However, these works tended to consider early WWW traffic whose profile differs significantly from today's Video-on-Demand traffic. As such, previously proposed models tended to laregly ignore the long-term popularity evolution of contents, in favour of capturing the short-time scale correlations of the contents request processes. Moreover, previous works have primarily focused on characterising the distribution of inter-requests times for a given content [15, 16], or on the distribu-

Table 1: The experiment traces.

Trace	PoP	Period (2012)	Length	IPs	Requests	Videos
TRACE 1	PoP 1	20/03-25/04	35 days	14224	1.7M	0.93M
TRACE 2	PoP 1	30/04-28/05	27 days	16172	1.8M	0.95M
TRACE 3	PoP 2	20/03-30/04	40 days	17242	2.4M	1.24M
TRACE 4	Pop 3	20/03-25/04	35 days	31124	3.8M	1.76M

tion of content requests that fall between two consecutive requests for a given content [14], providing only a partial characterisation of temporal locality. Further, in contrast to this work, previous works have not proposed a phenomenological model that captures the fundamental origins of temporal locality.

2. THE STANDARD APPROACH

The most common approach to evaluating the performance of caching systems is to assume that content requests are generated under the IRM, with a request distribution following a generalised Zipf law [6, 10, 11, 12, 18]. The IRM considers a fixed population of N contents, such that the sequence of content requests arriving at the cache is characterised by the following set of fundamental properties: the probability of a request for a given content n, for $1 \leq n \leq N$, is constant (i.e. the content's popularity does not vary over time) and independent of all past requests. Finally, the set of available contents does not change over time. While in its simplest form, Zipf's law states that the probability of requesting the nth most popular item is proportional to $1/n^{\alpha}$, where the exponent α depends on the considered system (especially on the type of contents) [2].

The IRM is widely adopted because of its simplicity and effectiveness in facilitating the development of tractable analytic models under various caching policies [19, 20, 21]. Although clearly, the assumptions made by the IRM are too rigid for real traffic, it has been advocated as an acceptable approximation, when, the popularity variations of contents over time are relatively slow, with respect to the time-scale of the content churn in the cache [10]. Whilst extensions of the IRM, aimed at capturing temporal correlations in the request process have been proposed in the literature, these tend to focus on applications other than video-on-demand [22, 23, 17]. Further, in contrast to the SNM, these works assume that the requests for each content are drawn from a fixed catalogue, and follow a stationary process (either a renewal process, or a Markov or Semi-Markov Modulated Poisson process). They therefore ignore the temporal evolution of content popularity, which plays a crucial role in the context of on-demand video.

In the rest of this section, we assess the applicability of the arguments for the IRM by critically discussing the common assumptions made when utilising it. Using real data traces, we show that when applied to cache performance analysis, the IRM assumptions lead to significant errors.

2.1 Data Set

In this work, we rely on passive measurements to characterise the popularity profiles of YouTube videos in operational networks. We employed Tstat [24], an open-source traffic monitoring tool developed at Politecnico di Torino, to analyse the packets exchanged by end-users from monitored vantage points. Tstat was installed on three PoPs (located in different large cities) of a large Italian ISP, connecting residential customers through ADSL and FTTH access technologies. Measurements were collected on both incoming and outgoing traffic carrying YouTube videos for a period of three months, from mid March 2012 to late May 2012. During this period, we observed the activity of more than 60,000 end-users accessing the Internet normally. The traces consider only TCP flows

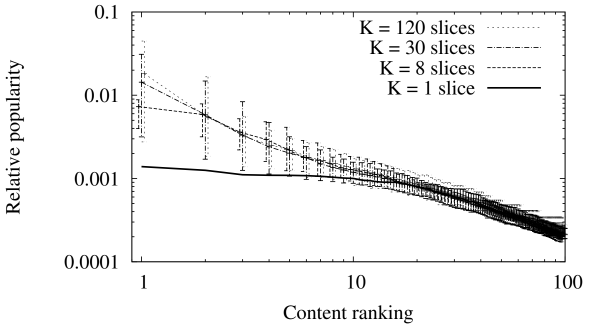

Figure 1: The empirical popularity distributions of the 100 most popular videos observed in TRACE 1. Each distribution is generated by splitting the trace into K slices, where "K=1" corresponds to the entire trace. For each K, we report the average of the relative frequency of each rank position across all the slices, and the error bars correspond to 5-95 percentiles.

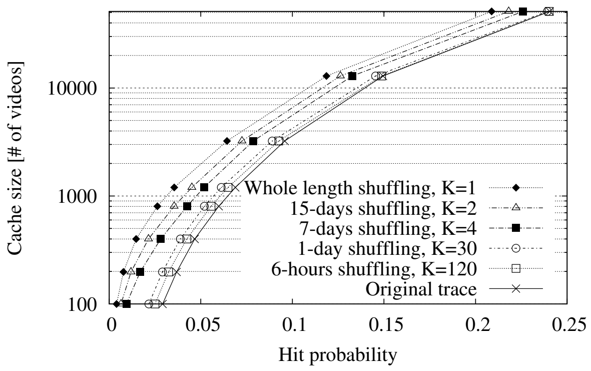

Figure 2: The cache size required to achieve a desired hit probability, when an LRU cache is populated by the requests contained in TRACE 1, subject to differing degrees of trace reshuffling. Note that $K = 1$ corresponds to the basic IRM.

corresponding to YouTube downloads. In total, we recorded almost 10 million transactions, accounting for approximately 227 TB of delivered content. Table 2 provides more details on the traces considered in our analysis; note that here, each IP is associated to one household. We verified that findings from TRACES 1-4 are general and not biased, since the results of their analysis are all almost identical.

2.2 Mistakes and Limitations of the IRM

The main difficulty that is encountered when adopting an IRM approach is the specification of the content popularity distribution, subject to a given content catalogue of size N. This distribution must be chosen with extreme care, since the cache performance depends heavily on the relative popularity of different contents [2, 15]. While the choice of a Zipf distribution (and its generalisations) is supported by experimental evidence, including measurements of web pages, file sharing, video-on-demand and user generated content access profiles, its parameterisation is not entirely obvious.

In fact, when determining a representative content request rate, given the availability of a long data trace, it is easy for practitioners to fall into the trap of directly plugging into the cache model, a popularity distribution estimated from long-term measurements. Even when it is reasonable to assume that cache dynamics are much faster than popularity variations (so that we can apply a time-scale separation), the instantaneous popularity distribution can differ significantly from the distribution derived from long-term measurement campaigns.

We use TRACE 1 to illustrate the impact of directly using the popularity distributions estimated from long-term measurements. To do so, we derive different empirical popularity distributions for the contents in the trace by first dividing it into K slices, such that each slice contains exactly the same number of requests. We then compute the relative popularity of the contents in each slice, with respect to the total number of requests observed in the slice (normalised to one). The results of this experiment are reported in Fig. 1, in which we plots, for each rank position of the 100 most popular videos, the average and the $5-95$ percentiles for the relative frequency of requests (evaluated across the K slices).

From Fig. 1, we observe that the steepness of the empirical distribution for the content popularity increases with respect to the number of slices considered. In practice, increasing K corresponds to having larger estimates of α for the Zipf distribution. Fitting the value of α for the tail of the distributions, we find that α ranges from 0.70 ($K = 1$) to 0.85 ($K = 120$). Moreover, as shown by the widening error bars, increasing K results in a higher variance in the relative popularity of the contents. The practical consequence of this observation is that estimates of the popularity distribution become more noisy as we decrease the time scale we consider (increasing K) . In short, it becomes increasingly more difficult to derive from the trace a reliable estimate of the popularity distribution as we consider decreasing time scales.

The result is that the error arising from using the long-term popularity distribution of contents, to predict cache performance (following the IRM approach), can be significant for a cache implementing the Least Recently Used (LRU) eviction policy (see Fig. 2). The reason for this can be understood when we consider that the above error is equivalent to that introduced by completely neglecting all temporal correlations (the so-called "temporal locality") in the arrival process of requests. To see this, suppose that we shuffle at random a trace of requests, the result would be that the long-term popularity distribution resulting from the shuffled sequence remains exactly the same, but all temporal correlations would be broken.

We could still pursue the IRM approach and correct for the loss in temporal correlation, however, this is not a straightforward task. To do so, we would first need to determine an "evaluation interval", that is comparable with the time-scale of cache dynamics, for the content popularity. Assuming that the popularity variations over the interval are negligible, we would then need to compute a modified popularity distribution for the contents requested in the interval. Notice that in doing so, we would need to compute it on a reduced catalogue of size much less than N, which is necessary to properly normalise the request probabilities relative to the interval.

Finally, to show the impact of the loss in temporal locality when evaluating the performance of a cache, in Fig. 2, we compare the cache size required to achieve a given cache hit rate, when (i) using the original trace (TRACE 1), and (ii) synthetic traces derived from the trace in which we control the degree of temporal locality in the trace. To construct the synthetic traces, we again partition the original trace into K slices, then randomly permute the requests within each slice to remove their temporal locality, such that, consecutive requests become independent.

It is worth noting that due to the limited duration of our traces, we are constrained to considering cache sizes such that the eviction time (i.e., the time since the last request after which a content is evicted from the cache) is significantly shorter than the duration of the whole trace. Under the request rates observed in our traces, the average eviction time is 2-3 days for cache size in the order of $50,000$ objects, which is the maximum considered in our experiments. This explains why in Fig. 2 we could only report hit probabilities in the range [0, 0.25].

From Fig. 2, the case where $K = 1$, corresponds to breaking all temporal correlations (i.e., the result of a naive application of the IRM approach). For instance, the sequence 11...1, 22...2, 33...3, which is clearly not independent would succeed most tests of independent sequence after shuffling with $K = 1$. By increasing K, we reduce the average size of the time window considered by each slice and obtain artificial traces that are increasingly more similar to the original trace, i.e., temporal correlations are broken only within slices of diminishing length. Fig. 2 shows two important findings: first, the IRM assumption leads to a significant over-estimation (by a factor between 2 and 10) of the required cache size, specially when the hit probability is low. Second, the impact of increasing K (i.e., decreasing the temporal duration of slices) is that the required cache size approaches the result for the unmodified trace, especially as the slice duration approaches the order of a few hours.

The results in Fig. 2 suggest that in this particular scenario, the IRM approach could produce accurate predictions of cache performance, provided that we are able to estimate the relative popularity of contents requested within intervals with a duration of around one day. However, this approach has several drawbacks:

- As shown in Fig. 1, due to the random fluctuations and possible long-range effects in the trace, it is in practice very hard to obtain from measurements, a content popularity distribution for short time-scales.
- It is even harder to check that popularity variations are negligible over the chosen interval, especially for the least popular contents, for which we will have fewer samples.
- In practice, the evaluation interval should be adapted to the time-scale of cache dynamics. This inevitably depends on several factors (aggregate requests arrival rate, cache size, caching policy, etc.), and forces users to recompute a different popularity distribution for each scenario.
- The time-scale of cache dynamics itself depends on the popularity distribution that we are trying to characterise, leading to a circular dependency.
- It is even possible (although only for very large caches and/or small request rates) that popularity variations cannot be considered negligible with respect to cache dynamics, rendering the IRM approach inaccurate.

With respect to the listed reasons, and in contrast to the common opinion, we believe that the IRM approach has severe limitations, especially in scenarios characterised by dynamic contents with evolving popularity profiles. The consequence is therefore a need for alternative approaches capable of recapturing the temporal locality in content requests. Before presenting our solution, we first take a closer look at the behaviour of content requests in our traffic traces.

3. TOWARDS A NEW MODEL

In this section we study the characteristics of the traffic traces with the goal of identifying the main factors that should be considered in order to describe the content request process.

There are two main factors responsible for the non-stationarity observed in real traffic: (i) the typical diurnal variation of the aggregate arrival rate of requests (see Fig. 3), and (ii) the fact that the arrival rate of requests for individual contents is highly non-stationary (see Fig. 4). We find that, contrary to common opinion [25], the diurnal variation has little impact on the main performance metrics for caches (e.g., hit probability, see Sec. 4.2). To understand why this is the case, consider that the hit probability of

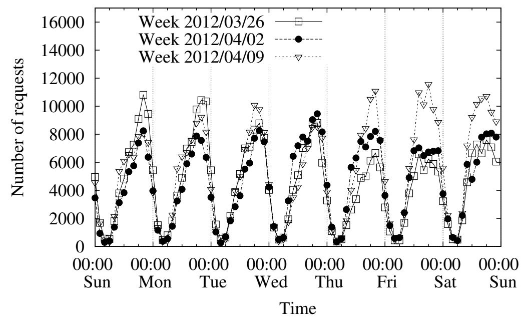

Figure 3: Evolution of the volume of requests over three weeks for TRACE 3. Requests here are binned in two-hour windows over the week.

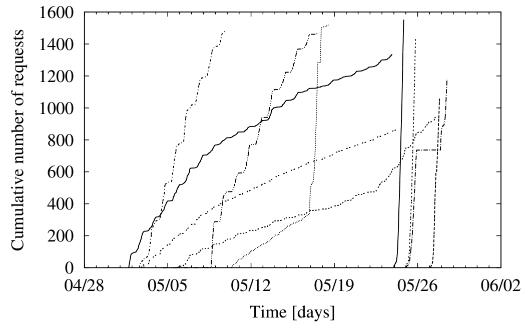

Figure 4: Cumulative number of requests over time for a subset of videos in TRACE 2; only the requests within the life-span of the content are shown..

almost all proposed caching policies depends only on the sequence of content IDs arriving at the cache, and not on the time-stamps associated with the requests. In other words, if we were to arbitrarily densify or dilute the sequence of content IDs over time, we would obtain the same hit probability.

With respect to the non-stationarity of content popularity, from Fig. 4, we see that YouTube videos display extremely heterogeneous request distributions and exhibit strong time-localities. For instance, we observe that the popularity of some videos vanishes after only a few days, while others continue to attract requests for almost the entire duration of the trace - reflecting the diversity in user interest. As a result, to capture the evolution of content popularity over time, we focus just on the this cause, and characterise each individual content object m with the following two parameters:

- The total number of requests (V_m) generated by the content.
- The effective life-span (l_m) of the content, which is defined as the duration of the interval in which we see the bulk of its requests[1].

However, because in practice we only see the requests arriving within a finite time window (i.e., the length of the trace), both of these quantities cannot be evaluated exactly from on-the-fly observations. We will therefore denote by $\hat{V}_m$ and $\hat{l}_m$ the estimated values of V_m and l_m, respectively, obtained by considering only the

[1]More precisely, the effective life-span (l_m) of an object is defined as the time elapsing between the two requests corresponding to $0.1V_m$ and to $0.9V_m$, respectively. The effective life-span permits us to filter out the impact of outliers on the average lifetime of an object (e.g., one more request arriving a long time after the previous requests).

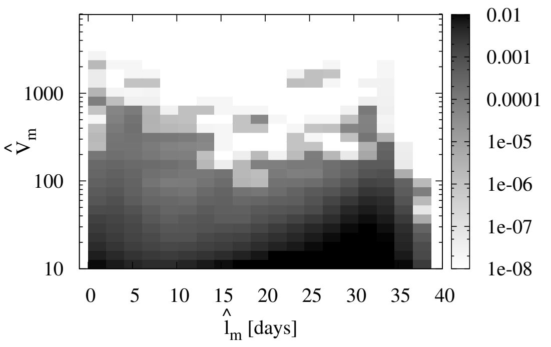

Figure 5: Density map of contents with $\hat{V}_m \geq 10$ observed in TRACE 3, based on estimated number of requests $\hat{V}_m$ and effective life-span $\hat{l}_m$.

requests appearing in the trace. We note that both variables tend to be underestimated with respect to their actual values, especially for contents whose true life-span is comparable to or greater than the trace length.

The density map in Fig. 5 reveals that, as expected, contents exhibit wide heterogeneity in terms of estimated life-spans $\hat{l}_m$ and estimated volumes $\hat{V}_m$. The map also shows that there exists a peculiar correlations between $\hat{l}_m$ and $\hat{V}_m$. This suggests that a traffic model should consider the joint distribution of these metrics. In fact, from the results, we observe that a non-marginal share of videos ($7 - 10\%$) exhibit a very small life-span ($\hat{l}_m \leq 5$ days), while 2% of videos have $\hat{V}_m \geq 10$, but account for a share of requests that is greater than 27% (these results hold for all the traces in our data set). These two observations show that a precise traffic generator should accurately model videos with short life-span, while accounting for the largest share in requests generated.

At this point, it is worth emphasising that the two parameters V_m and l_m alone do not completely characterise the temporal evolution of content popularity, which as shown in Fig. 4, can exhibit complex growth patterns. In fact, recent studies [26, 27, 28] conducted on much larger data sets reveal that the popularity of different contents, including videos, follow a limited number of "archetypal" temporal profiles, which essentially depend on the nature of the content and on the way it becomes popular. However, as we will see in Sec. 4.2, cache performance is essentially driven by the parameters V_m and l_m, while the shape of the popularity profile has only a second-order effect. Therefore, a more accurate representation of popularity evolution, in terms of detailed temporal profiles, is not strictly necessary in order to accurately predict cache performance. The practical side effect of this observation is that we can design a simple, accurate and robust model to reason about the temporal evolution of content popularity.

4. SHOT NOISE TRAFFIC MODEL

Given the observations in the previous section, we now turn to proposing a novel approach to describing the arrival process of content requests generated by a large population of users. The goal here is to maintain the generality and flexibility of the IRM approach, but improve on its accuracy without significantly increasing the model's complexity. In more detail, the model must: (1) be general and flexible; (2) provide a native explanation for the temporal locality of the request process; (3) explicitly represent content popularity dynamics; (4) capture the phenomena that have a major impact on cache performance while ignoring those with no or limited impact; (5) be as simple as possible while maintaining ac-

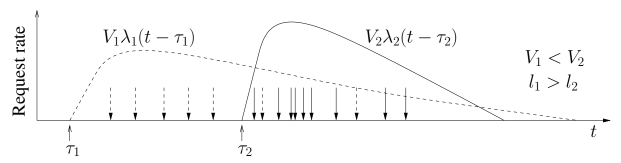

Figure 6: Example of requests (denoted by arrows) generated by two contents with different catalogue insertion times (τ_1, τ_2), average number of requests (V_1, V_2) and profiles ($\lambda_1(t)$,$\lambda_2(t)$).

curacy; (6) enable users to analytically investigate the performance of popular caching policies.

Our proposed solution is to represent the overall request process as the superposition of many independent processes, each referring to an individual content. In particular, each content m is characterised by three physical parameters (τ_m, V_m, $\lambda_m(t)$): τ_m represents the time instant at which the content enters the system (i.e., when it can be requested by the users); V_m denotes the (average) number of requests generated by the content; and $\lambda_m(t)$ is the "popularity profile", describing how the request rate for a given object m evolves over time. In general, $\lambda_m(t)$ is defined to be a function satisfying the following conditions: (positiveness) $\lambda_m(t) \geq 0$, $\forall t$; (causality) $\lambda_m(t) = 0$, $\forall t < 0$; (integrability and normalisation) $\int_0^\infty \lambda_m(t)\,\mathrm{d}t = 1$.

Given the above parameters, our model assumes that the request process for a given content m is described by a time-inhomogeneous Poisson process whose instantaneous rate at time t is given by:

$$V_m \lambda_m(t - \tau_m)$$

For the sake of simplicity, we assume that new contents become available in the system according to a homogeneous Poisson process of rate γ, i.e., time instants $\{\tau_m\}_m$ form a standard Poisson process. We refer to this model as Shot Noise Model (SNM), since the overall process of requests arrival is known as a Poisson shot-noise process [29]. Fig. 6 illustrates an example of the request pattern generated by the superposition of two "shots" corresponding to two contents with quite different parameters.

We emphasise that the above Poisson assumptions on the instantaneous generation process of requests for each content, and on the arrival process of new contents, are essentially introduced for the sake of analytical tractability. However, they are very well justified by the experience gained from our traces. In fact, the results in Fig. 2 suggest that we need not be very concerned with temporal correlations in very short time-scales (in the order of a few hours, say less than 6): removing all correlations at very short time scales (as given in Fig. 2) has no significant impact on the resulting cache performance. The practical consequence is that we do not need to take into account possibly complex correlations in the arrival process of requests such as might be induced by popularity cascades [27, 30, 31]. While cascades in popularity are indeed observed in large catalogues with a geographically distributed user base, this is not very evident in our traffic traces, which are much more local. Given that short time-scale correlations (up to a few hours) are not important (in contrast to the scenarios considered in [27, 30]), the adoption of a Poisson model for the arrival process of requests is well justified, and enables us to build simple analytic models for cache performance analysis such as those developed in [32].

For each given content, the SNM requires that users specify its entire popularity profile in the form of the function $\lambda_m(t)$, which, given the difficulty in estimating popularity profiles from a trace, could be considered as a limitation. However, we have found that it is not necessary to precisely identify the shape of $\lambda_m(t)$. In fact, a simple first-order approximation, according to which we just specify the content life-span l_m, is enough to obtain accurate predictions of the performance of the cache. In other words, we can arbitrarily choose any reasonable function $\lambda_m(t)$ with an assigned life-span l_m, and obtain almost the same results in terms of cache performance (see Fig. 7). Finally, content heterogeneity is taken into account by associating a life-span l_m with every content, jointly with the (typically correlated) total number of requests V_m. This means that, upon arrival of each new content m, independently for each content, we randomly choose the pair of parameters (V_m, l_m) from a given assigned joint distribution.

Table 2: Model parameters for content classes 1–5.

Class	Life-span [days]	Trace	%Reqs	%Videos	$E[\hat{l}_m]$	$E[\hat{V}_m]$
Class 1	$\hat{l} \leq 2$	TRACE 1	9.15	3.17	1.14	86.4
		TRACE 2	10.05	4.17	1.09	76.2
		TRACE 3	9.44	3.73	1.04	76.0
		TRACE 4	7.77	3.34	1.06	74.0
Class 2	$2 < \hat{l} \leq 5$	TRACE 1	6.80	4.9	3.36	41.9
		TRACE 2	12.55	7.83	3.34	50.7
		TRACE 3	6.55	4.54	3.32	43.3
		TRACE 4	6.12	4.06	3.41	48.0
Class 3	$5 < \hat{l} \leq 8$	TRACE 1	5.87	2.95	6.40	59.5
		TRACE 2	6.72	4.74	6.31	44.9
		TRACE 3	6.05	2.87	6.42	63.3
		TRACE 4	5.14	2.71	6.45	60.3
Class 4	$8 < \hat{l} \leq 13$	TRACE 1	5.49	4.45	10.53	36.9
		TRACE 2	10.79	8.61	10.86	39.6
		TRACE 3	4.84	3.68	10.62	39.5
		TRACE 4	5.34	4.48	10.65	37.8
Class 5	$\hat{l} > 13$	TRACE 1	72.69	84.58	24.61	25.7
		TRACE 2	59.89	74.65	19.29	25.3
		TRACE 3	73.11	85.17	28.19	25.8
		TRACE 4	75.63	85.41	24.59	28.1

4.1 Parameter Fitting

There are many ways to derive the parameters of the SNM from a trace. In this section, we present the simple approach which we have employed to fit our data traces. We first partition the contents into 6 classes $(0, \ldots, 5)$, on the basis of the measured request volume $\hat{V}_m$ and content life-span $\hat{l}_m$. Content Class 0 is defined to contain objects with $\hat{V}_m < 10$, representing contents for which we cannot derive a reliable estimate of the life-span, because they gather too few requests. Due to this, we treat contents in this class using the IRM approach, and assume that their requests fall uniformly at random within the synthesised trace. It is worth acknowledging that, in doing this, we lose an opportunity to characterise the time locality of a significant fraction of contents (85% of the contents have $\hat{V}_m < 10$). However, even with this restriction, as we will see, we can still obtain a reasonable conservative prediction on the resulting cache performance, (i.e., we slightly overestimate the cache size required to achieve a desired hit ratio).

Classes 1 to 5 contain contents with $\hat{V}_m \geq 10$ and, as shown in Table 2, are partitioned based on their estimated life-spans ($\hat{l}_m$). For each content class, Table 2 reports the percentage of total requests attracted by the class, the percentage of contents belonging to it, and their average estimated values $\hat{V}_m$ and $\hat{l}_m$. Notice from Table 2 that contents in Class 1, whose life-span is smaller than 2 days, represent less than 4% of the total number of contents but attract approximately 10% of all requests. Therefore, because these contents exhibit a large degree of temporal locality, they can be expected to have significant impact on cache performance.

Observe also from Table 2 that the values related to each class are quite similar across the four traces (within a factor of 2). This is significant, because it suggests that our broad classification captures some invariant properties of the considered traffic. Therefore,

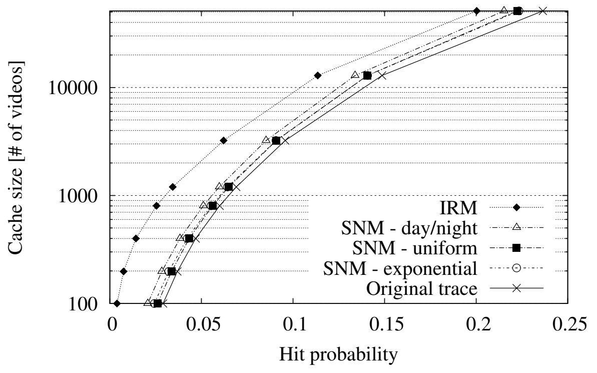

Figure 7: Cache size vs hit probability under LRU policy for TRACE 4. Note that similar results are obtained for all traces.

we have the opportunity of building a synthetic traffic model that is fairly general and flexible, by fitting the parameters that appear to be invariant (or almost invariant) from one single trace, and properly scaling those parameters that clearly depends on the particular context (such as the number of end-users).

However, because our traces cover a period of approximately one month each, they are not long enough to extract very general laws, especially for long-lived contents. This is particularly evident for contents in class 5 ($\hat{l}_m > 13$ days), whose life-span is comparable with the trace length. Therefore, their estimated value of $\hat{l}_m$ is expected to be strongly affected (i.e., underestimated) by border effects due to the trace being finite. For this reason, to obtain a conservative prediction, we again treat contents in this class as if they had a stationary popularity (like in the IRM), generating their requests uniformly in the considered time horizon.

In sum, given the six broad classes of behaviour defined, only objects in Classes 1 to 4 (see Table 2) are generated according to the SNM model as follows. First, based on the estimated rate at which contents are generated in each class (see column "Videos" in Table 2), we sample the time instants $\{\tau_m\}$ at which contents of the class become available. This can be done in a standard way, because the arrival process of contents within each class is assumed to be homogeneous Poisson. Second, we set the life-span of all the contents of the class equal to the corresponding estimated average life-span $E[\hat{l}_m]$, thereby compensating for border effects due to content life-spans comparable with the trace duration. Third, we randomly choose V_m for each content generated in the class according to the corresponding empirical distribution observed in the trace. Finally, the events corresponding to content requests are generated for each content m according to an inhomogeneous Poisson process with fertility function $V_m\lambda(t - \tau_m)$ according to standard methods [33]. In Sec. 4.3, we will discuss the computational cost associated with the SNM.

Moreover, for objects in Classes 1 to 4 we have chosen a unique "shape" ($\lambda_m(t)$) for the popularity profile of all contents and considered two different shapes, with parameter L (content lifetime) as: (1) exponential popularity $\lambda(t) = \frac{1}{L}e^{-\frac{t}{L}}$ for $t \geq 0$; (2) uniform popularity $\lambda(t) = \frac{1}{2L}$ for $t \in [0, 2L]$. Note that, for each content class, L is chosen to match the desired life-span $E[\hat{l}_m]$ [2].

4.2 Model Validation

To evaluate the accuracy of our traffic model we generate synthetic request traces as described in Sec. 4.1 and feed them to a cache implementing the LRU policy. We again report the cache size required to achieve a desired hit probability. For comparison, we also report the results obtained with the original trace and its completely shuffled version. The latter neglects the contents' temporal locality and provides the same performance as the IRM approach; based on long-term measurements of the content popularity distribution.

From Fig. 7 we find that the results obtained using the SNM (using either the uniform or the exponential shape) are very close to those obtained with the original trace. As mentioned, the shape of the popularity profile chosen (exponential/uniform) has little impact on the results. While in contrast, the discrepancies produced by the shuffled trace (IRM) are quite large, especially for small caches. This confirms that the impact, on cache performance, of a small number of highly popular contents with a relatively short life-span (in the order of a few days) is significant and should not be neglected.

The results in Fig. 7 show that the SNM provides an accurate prediction of cache performance, despite the heavy simplifications adopted in the parameter identification. We expect that even better predictions could be achieved by improving the fitting procedure, or, if available using much longer traces.

Finally, Fig. 7 also reports (label Day/Night) an extended version of the model that incorporates the effects of daily oscillations in the traffic rate. This is produced by simply modulating the shots of all contents by a fixed, periodic function of time $f(t) = 1 + \sin(2\pi t)$ (where t is expressed in days). From this result, we see that the daily variations have a marginal impact on the cache performance.[3]

4.3 Computational cost

Due to its simplicity, the SNM can be effectively employed to generate synthetic traces for large-scale simulations, permitting users to explore realistic scenarios that could hardly be studied by employing experimental traces alone. More precisely, the computational cost to generate a synthetic trace using the SNM is only slightly larger than the cost for the IRM. When a new content with parameters V_m and l_m is introduced to the system, we can first generate the total number of requests received by this content according to a Poisson distribution of mean V_m, then schedule ahead of time; the arrival time of each request according to the content's temporal profile (which depends on l_m), recording it into an ordered or partially ordered data structure (e.g., a heap), to be dynamically used in an event-driven simulator.

Therefore, with respect to the $\Theta(1)$ cost needed to generate a request under the IRM, the cost to generate a request under the SNM equals the insertion time of an element into an ordered data structure; which is $\Theta(\log M)$, where M is the number of requests scheduled in the future. Notice that, whilst in the worst case we can take M equal to the trace length. In practice M is much smaller than this, since it is related to the average number of requests falling in a shot, i.e., $M = \Theta(\gamma\,\mathbb{E}(V_m\, l_m))$. From our experience, the additional time needed to simulate a cache under our synthetic traffic model, compared to the time needed when an experimental trace is available, is marginal.

5. FINAL CONSIDERATIONS

This work, to the best of our knowledge, presents the first quantification of the error arising from applying the IRM to the evalu-

[2]For instance, in the case of the uniform shape, $L = \frac{0.5E[\hat{l}_m]}{0.8}$.

[3]The adopted modulation technique does not exactly produce the effect of just stretching/squeezing the sequence of requests generated by the un-modulated process, which explains why the impact on cache performance is not null.

ation of cache performance in the context of on-demand video delivery. We have shown that the error that the IRM induces is large enough to render it too pessimistic for practical use, especially in scenarios with small cache sizes. As a replacement, we proposed a parsimonious and novel approach termed the Shot-Noise Model (SNM) that accurately models cache performance while being sufficiently simple to be effectively employed in both analytical and scalable simulative studies of large caching systems.

The SNM provides a flexible and accurate approach to model and synthesise contents' request arrival process and does so by capturing some of the fundamental characteristics of temporal locality. This paper provides an accurate tool to investigate and understand the performance of caches in content-driven systems. Having such a tool enables the development of accurate forecasts of the network's performance, leading to a more efficient dimensioning of content delivery systems and ultimately reducing operating costs while improving the experience of end-users.

6. ACKNOWLEDGMENTS

We thank Marco Mellia, Felipe Huici, Alessandro Finamore and the anonymous reviewers for providing useful feedback on earlier drafts of this paper. This research is supported by funding from the European Union under the FP7 Grant Agreement no. 318627 (Integrated Project "mPlane").

7. REFERENCES

[1] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard. Networking named content. In *ACM CoNEXT*, 2009.

[2] C. Fricker, P. Robert, J. Roberts, and N. Sbihi. Impact of traffic mix on caching performance in a content-centric network. In *NOMEN Workshop*, 2012.

[3] J. Erman, A. Gerber, M. Hajiaghayi, D. Pei, S. Sen, and O. Spatscheck. To cache or not to cache: the 3G case. *IEEE Internet Computing*, 15(2), March 2011.

[4] F. Qian, K. S. Quah, J. Huang, J. Erman, A. Gerber, Z. Mao, S. Sen, and O. Spatscheck. Web caching on smartphones: ideal vs. reality. In *ACM MobiSys*, 2012.

[5] U. Lee, I. Rimac, D. Kilper, and V. Hilt. Toward energy-efficient content dissemination. *IEEE Network*, March 2011.

[6] W. Jiang, S. Ioannidis, L. Massoulié, and F. Picconi. Orchestrating massively distributed CDNs. In *ACM CoNEXT*, 2012.

[7] I. Poese, B. Frank, G. Smaragdakis, S. Uhlig, A. Feldmann, and B. Maggs. Enabling content-aware traffic engineering. *ACM SIGCOMM CCR*, Sep. 2012.

[8] J. Erman, A. Gerber, M. T. Hajiaghayi, D. Pei, and O. Spatscheck. Network-aware forward caching. In *ACM WWW*, 2009.

[9] E. Coffman and P. Denning. *Operating Systems Theory.* Prentice-Hall, Englewood Cliffs (NJ), 1973.

[10] C. Fricker, P. Robert, and J. Roberts. A versatile and accurate approximation for LRU cache performance. In *ITC*, 2012.

[11] B. Ager, F. Schneider, J. Kim, and A. Feldmann. Revisiting cacheability in times of user generated content. In *IEEE Global Internet Symposium*, March 2010.

[12] N. Laoutaris, G. Zervas, A. Bestavros, and G. Kollios. The cache inference problem and its application to content and request routing. In *IEEE INFOCOM*, 2007.

[13] S. Vanichpun and A. M. Makowski. The output of a cache under the independent reference model: where did the locality of reference go? *ACM SIGMETRICS Perform. Eval. Rev.*, Jun. 2004.

[14] V. Almeida, A. Bestavros, M. Crovella, and A. de Oliveira. Characterizing reference locality in the WWW. In *IEEE PDIS*, 1996.

[15] S. Jin and A. Bestavros. Sources and characteristics of Web temporal locality. In *IEEE/ACM Mascots*, 2000.

[16] R. Fonseca, V. Almeida, M. Crovella, and B. Abrahao. On the intrinsic locality of Web reference streams. In *IEEE INFOCOM*, 2003.

[17] P. R. Jelenković and A. Radovanović. Least-recently-used caching with dependent requests. *Theoretical computer science*, 326(1):293–327, 2004.

[18] E. Rosensweig, D. Menasche, and J. Kurose. On the steady-state of cache networks. In *IEEE INFOCOM*, 2013.

[19] P. R. Jelenković and A. Radovanović. The persistent-access-caching algorithm. *Random Struct. Algorithms*, 33(2):219–251, Sept. 2008.

[20] A. Dan and D. Towsley. An approximate analysis of the LRU and FIFO buffer replacement schemes. In *ACM SIGMETRICS*, 1990.

[21] H. Che, Y. Tung, and Z. Wang. Hierarchical Web caching systems: modeling, design and experimental results. *IEEE JSAC*, 20(7), Sept. 2002.

[22] E. G. Coffman and P. J. Denning. *Operating systems theory*, volume 973. Prentice-Hall Englewood Cliffs, NJ, 1973.

[23] K. Kylkoski and J. Virtamo. Cache replacement algorithms for the renewal arrival model. In *Fourteenth Nordic Teletraffic Seminar, NTS-14*, pages 139–148, Copenhagen, Denmark, Aug. 1998.

[24] A. Finamore, M. Mellia, M. Meo, M. M. Munafò, and D. Rossi. Experiences of Internet traffic monitoring with Tstat. *IEEE Network*, 2011.

[25] H. Abrahamsson and M. Nordmark. Program popularity and viewer behaviour in a large tv-on-demand system. In *ACM IMC*, 2012.

[26] J. Yang and J. Leskovec. Patterns of temporal variation in online media. In *WSDM '11*, 2011.

[27] Y. Matsubara, Y. Sakurai, B. A. Prakash, L. Li, and C. Faloutsos. Rise and fall patterns of information diffusion: model and implications. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 6–14. ACM, 2012.

[28] M. Ahmed, S. Spagna, F. Huici, and S. Niccolini. A peek into the future: Predicting the evolution of popularity in user generated content. In *ACM WSDM*, Feb. 2013.

[29] J. Møller. Shot noise Cox processes. *Advances in Applied Probability*, 35(3), 2003.

[30] R. Crane and D. Sornette. Robust dynamic classes revealed by measuring the response function of a social system. *Proceedings of the National Academy of Sciences*, 105(41):15649–15653, 2008.

[31] M. Cha, A. Mislove, and K. P. Gummadi. A measurement-driven analysis of information propagation in the Flickr social network. In *WWW '09*, 2009.

[32] M. Ahmed, S. Traverso, P. Giaccone, E. Leonardi, and S. Niccolini. Analyzing the performance of LRU caches under non-stationary traffic patterns. *CoRR*, abs/1301.4909, 2013.

[33] S. M. Ross. *Simulation*. Elsevier Academic Press, Amsterdam, 2006.