An Inverse Problem Approach for Content Popularity Estimation

Felipe Olmos
Orange Labs and CMAP, École Polytechnique*
luisfelipe.olmosmarchant@orange.com

Bruno Kauffmann
Orange Labs
bruno.kauffmann@orange.com

ABSTRACT

The Internet increasingly focuses on content, as exemplified by the now popular Information Centric Networking paradigm. This means, in particular, that estimating content popularities becomes essential to manage and distribute content pieces efficiently. In this paper, we show how to properly estimate content popularities from a traffic trace.

Specifically, we consider the problem of the popularity inference in order to tune content-level performance models, e.g. caching models. In this context, special care must be taken due to the fact that an observer measures only the flow of requests, which differs from the model parameters, though both quantities are related by the model assumptions. Current studies, however, ignore this difference and use the observed data as model parameters. In this paper, we highlight the inverse problem that consists in determining parameters so that the flow of requests is properly predicted by the model. We then show how such an inverse problem can be solved using Maximum Likelihood Estimation. Based on two large traces from the Orange network and two synthetic datasets, we eventually quantify the importance of this inversion step for the performance evaluation accuracy.

Categories and Subject Descriptors

C.2 [**Computer-Communication Networks**]: General

Keywords

Popularity Distribution, Mixture Model, Maximum Likelihood Estimation, Performance models, Caching

1. INTRODUCTION

"Content is king", says nowadays a popular Internet meme. This advent of ubiquitous content is reflected on the Internet, both by the importance of Content Distribution Networks (CDNs) and transparent caching for coping with an ever-increasing traffic demand, and by the emergence of the Information Centric Networking (ICN) paradigm. Understanding content and, in particular, its popularity is now essential to improve the Internet and its applications. Content-level performance models are therefore a key tool in the analysis, design and dimensioning of networks.

Sparse models are particularly useful, since they capture the salient features of the system while remaining simple enough for analysis, depending only on a few parameters. These parameters have a large impact on the model output; yet one cannot observe them directly in measurements. Carrying a sensible analysis using the chosen model therefore requires solving the *inverse problem* to find the best model parameters of the system from the measurements.

Due to the rise of content, the number of available documents and their popularity distribution are now key parameters for traffic models. They have attracted significant attention from the community in the context of user generated content [3, 9], HTTP traffic [10, 13], and peer-to-peer networks [4, 19]. However, the measurement methods used in these works are not suited for parameterizing a performance model. In fact, they fail to take into account that the request count for a given document in a given observation period, within the framework of a stochastic model, is not a fixed value, but a random variable. In particular, they ignore the fact that, in traffic traces, objects with no request are not observed, being thus a *zero-censored* sample.

Our main objective in this paper is to provide a sound methodology for popularity estimation, with the aim of correctly fitting performance models. This requires to take into account the stochastic relation between the model parameters and the request counts that are observed in a given dataset. To this aim, we follow [4] in constructing Maximum Likelihood (ML) estimates. We illustrate the aforementioned issues and methodologies in the case of Poisson based traffic models in the context of caching performance. Nonetheless, the essential paradigm that we propose is applicable to other traffic models and contexts. Note that the choice of relevantmodels is outside the scope of this paper.

The rest of this paper is organized as follows. We first review the literature in Section 2, and describe in Section 3 the datasets we use. We then explicitly identify and formulate in Section 4 the inverse problem that consists in correctly calibrating performance models from trace measurements. To our knowledge, such a formulation has not been provided in previous studies. In Section 5, we propose a ML estimation method for this inverse problem. Section 6 provides a numerical evaluation of our approach. We discuss our results

*Centre de Mathématiques Appliquées, École Polytechnique, CNRS, Université Paris-Saclay.

.

VALUETOOLS 2015, December 14-16, Berlin, Germany

DOI 10.4108/eai.14-12-2015.2262621

and possible extensions in Section 7.

2. RELATED WORK

The works we here review falls into two broad categories: content popularity estimation from traffic measurements and statistical methods for mixtures models.

Due to the fact that popularity distributions usually exhibit a power law behavior, a common method to estimate them is to fit its rank-frequency distribution in double logarithmic scale. This approach has been recently criticized by Clauset et al. [4]. The main issue is that the rank-frequency plot is not a reliable statistic since, for example, it can exhibit power-law behavior even if the ground-truth does not.

Despite these problems, the use of the latter method is still pervasive in performance evaluation [8, 11] and traffic characterization studies [12, 13, 2]. Authors try to improve these methods by means of various adjustments. In [13], for example, authors separate in three parts the rank-frequency plot adjusting different curves in each piece and in [12], authors adjust "stretched exponential" curves instead of power-laws.

The latter adjustments indeed solve some of the fitting issues. In previous studies [11, 17], we have noted another issue in the context of performance models, which arises from the fact that it is permitted to objects to have zero request. In consequence, from the point of view of the network operator, objects with no request are not observed in traces. In the statistical jargon, this is called *zero-censored* and not taking this fact into account leads one to underestimate the catalog size, which has an impact on the conclusions drawn from the fitted model (see Section 6).

In the present work, we address the previous issues by using ML estimates. This method allows us to seamlessly handle the zero-censored case and it is proposed by Clauset et al. [4] as a robust method to fit heavy tailed data, which is a common property in popularity distributions. Maximum likelihood methods have already been in use for flow size estimation [16] and call center modeling [18]. The latter work uses an approach similar to ours, but it is limited to a specific parametric model for non-censored data. More importantly, our work highlights the fact that the assumptions of the performance model must be taken into account for a proper popularity estimation.

The statistical basis of our methods is the estimation of mixed discrete distributions, a subject that has been extensively studied in the literature. The non-parametric case has been addressed from two points of view: the first one searches the mixing density in the space generated by Laguerre polynomials with an exponential cut-off; the estimator is then obtained by a projection on the latter space [20, 5]. It, however, converges slowly with the sample size unless the density belongs to the aforementioned space. We therefore base our methodology on the second point of view, which assumes the mixing distribution to be a sum of Dirac masses. The estimation methods are then similar to an Expectation-Maximization scheme (EM) [15]. As regards the parametric case, EM schemes for finding the parameters of the mixing distribution are provided for many families in [14]. In both parametric and non-parametric cases, the estimation algorithms do not handle the case of censored data, and thus we simply use an all-purpose nonlinear optimization solver to obtain our results.

3. DATASETS

We base our analysis on two real-traffic datasets, called `#yt` and `#vod` respectively. Dataset `#yt` comes from the YouTube traffic delivered for three months in 2013 by the Orange Network in Tunisia, while `#vod` comes from the Video-on-Demand Orange service in France for 3.5 years. The traffic consists in 46 000 000 (resp. 3 400 000) requests to 6 300 000 (resp. 120 000) videos in the `#yt` (resp. `#vod`) set. More details on the collection and processing of these two datasets can be found in [17].

We also use two synthetic datasets, called `#prt` and `#delta`. This allows us to highlight in a more clear way some of our findings and, more importantly, to validate the results with controlled experiments when the ground-truth is not available. The set `#prt` (resp. `#delta`) is generated by first drawing 10 000 000 (resp. 100 000) random samples with distribution Pareto $(1.6, 0.1)$ (resp. Dirac delta at 4.0) representing the popularity (see section 5.1 for a model description). The number of requests for each document is then drawn according to the Poisson distribution with mean equal to the document popularity. After discarding the documents with zero request, this results into 2 600 000 (resp. 400 000) requests to 1 900 000 (resp. 98 000) documents.

4. PROBLEM DEFINITION

In the following, we are given a stochastic object-level model predicting some performance indicator. The predicted performance explicitly depends on a few parameters which characterize each object (e.g., document popularities, lifespans, sizes). It also strongly depends, however, on some implicit assumptions about the traffic or request process.

An example of such a situation is the evaluation of the hit ratio of a Least Recently Used (LRU) Cache, which is typically performed using the Independent Reference Model (IRM). In this context, users request documents among a catalog of K documents. These requests are intercepted by a cache server, which can store and serve only an evolving subset of the catalog. The IRM assumes that the sequence of requests for document $1 \leq k \leq K$ is a Poisson process with intensity λ_k, where λ_k is proportional to the popularity of document k; all such processes are mutually independent and their superposition build up the total request process. In this model, the number N_k of requests for document k in a time window W is an independent Poisson random variable $\mathcal{P}(\lambda_k W)$ of mean $\lambda_k W$. Up to a time normalization, we assume in the following that $W = 1$.

Figure 1 illustrates those two stages, both for an arbitrary performance model and the IRM case. The first stage consists in mapping the model parameters to a request flow (or a request flow distribution). The second step of the model computes the performance indicator, based on this request flow. In order to keep this paper concise, we now limit ourselves to the IRM model (see Section 7.1 for extensions).

Assume now that an observer has access to a sample of the actual request flow, e.g., a trace dataset or server logs. In the case of IRM, a sufficient statistics of the request process is the request counts $n_1, n_2, \ldots, n_{K_0}$ for all observed document, where K_0 is the number of observed documents in the sample. Following the point of view of an Internet Service Provider (ISP), we here assume that objects with zero request *are not observable* in the sample. Our main objective is to solve the following inverse problem: *estimate the popularity distribution such that the request flow predicted by the*

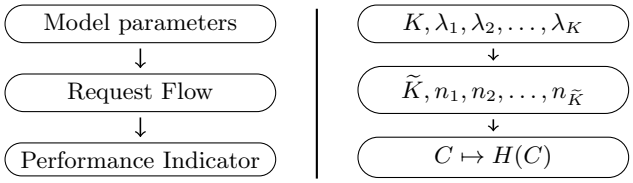

Figure 1: Schematic view of a performance model (left); example in the IRM case (right)

model using these parameters represents the data at best.

A simple solution, henceforth called the *naive method*, is to estimate the popularity of a document by its request count and the catalog size by the number of observed objects, that is: $\hat{K}^{\text{nv}} = K_0$ and $\hat{\lambda}_k^{\text{nv}} = n_k$, for $1 \le k \le \hat{K}^{\text{nv}}$.

We identify two problems at this stage. First, since the trace is zero-censored, with high probability the observed number of documents K_0 is strictly smaller than the catalog size K. Second, each document popularity λ_k is estimated by a single sample n_k of the random count N_k. This last limitation is well illustrated in the case of the `#delta` dataset. By definition, the ground-truth (real) popularities are $\lambda_k = 4$. In the dataset, however, the counts of document requests are Poisson random variables of mean 4, hence $\hat{\lambda}_k^{\text{nv}} = \mathcal{P}(4)$ and the naive estimation "dilutes" the mass of popularities over the set of positive integers. In Figure 2, we

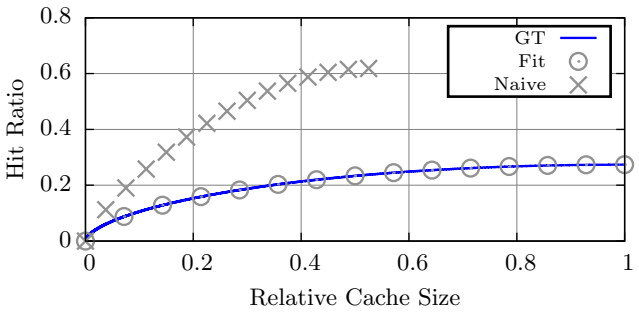

Figure 2: *Hit ratio of a cache fed by* `#prt` *trace: ground-truth (GT) and prediction by the naive estimation. The cache size is normalized with respect to that of the GT.*

show the impact of these limitations for the hit ratio estimation, based on the `#prt` trace. The first curve is our ground-truth. It is obtained via simulation of a LRU cache starting from an empty cache; the cache is fed by the traffic trace that is randomly shuffled to enforce the IRM assumption. The second curve is the prediction of the IRM model, when fed by the real popularities in the trace (see Section 8.1 for a quick derivation of the transient hit ratio for the IRM). As expected, it perfectly fits the ground-truth. The third curve shows the results obtained by the IRM model when fed by the parameters $\hat{K}^{\text{nv}}$ and $\hat{\lambda}_k^{\text{nv}}, 1 \le k \le \hat{K}^{\text{nv}}$, from the naive estimation. The hit ratio curves are seen to clearly differ, and the naive method proves inaccurate for estimating document popularities when fitting a performance model.

In the absence of any prior knowledge about the popularity distribution, the only available data for the estimation of each document popularity is a single request count, which limits the accuracy of this approach. To overcome this lack of information, we thus aim at jointly estimating the set of popularities, from the joint set of request counts. The latter approach allows us to use all the information contained in the joint Poisson distribution rather than just the mean.

Our problem can therefore be stated as follows:

Problem Statement: *Given the measured request counts* $\{n_1, n_2, \ldots, n_{K_0}\}$*, determine the parameters* $\hat{K}$ *and* $\hat{\lambda}_1$*,* $\hat{\lambda}_2$*,* $\ldots, \hat{\lambda}_{\hat{K}}$ *so that the set of random variables* $\{N_1, N_2, \ldots, N_{\hat{K}}\}$*, where* $N_k = \mathcal{P}(\hat{\lambda}_k)$ *for* $1 \le k \le \hat{K}$*, is the "closest" to the set* $\{n_1, n_2, \ldots, n_{K_0}, 0, \ldots, 0\}$*, with* $\hat{K} - K_0$ *zeros at the tail.*

5. MAXIMUM LIKELIHOOD ESTIMATION

In this section, we show how to solve the latter inverse problem via the Maximum Likelihood method.

In the IRM setting, the parameters $(\lambda_1, \lambda_2, \ldots, \lambda_K, K)$ are not ordered, and thus every request count could correspond to any of the popularities. The likelihood given observations $n = (n_1, n_2, \ldots, n_K)$ thus runs through every permutation σ of size K. Specifically the likelihood $\mathcal{L} = \mathcal{L}(\lambda_1, \lambda_2, \ldots, \lambda_K, K; n)$ is given by

$$\mathcal{L} = \frac{1}{K!} \sum_{\sigma} \left(\prod_{j=1}^{K_0} \frac{e^{-\lambda_{\sigma(j)}} \lambda_{\sigma(j)}^{n_j}}{n_j!} \times \prod_{j=K_0+1}^{K} e^{-\lambda_{\sigma(j)}} \right).$$

This combinatorial explosion for large K makes the ML method intractable for the IRM model. We thus propose in the following a slightly modified model, which is simultaneously tractable for ML estimations and simple to analyze.

5.1 IRM Mixture Model (IRM-M)

In order to succinctly describe the popularity parameters $\lambda_1, \lambda_2, \ldots, \lambda_K$ and to ease their estimation, we slightly modify the IRM model by considering them as random variables. Specifically, we assume that $\lambda_1, \lambda_2, \ldots, \lambda_K$ are an i.i.d. sample from an unknown *mixing distribution* with density g. Given the value of λ_k, the request process to the k^{th} document remains a Poisson process of intensity λ_k, and thus the counts of each document follow a mixed Poisson distribution with mixing distribution g. In particular, the number of requests N for any document satisfies

$$\mathbb{P}[N = j] = \mathbb{E}_g \left[\frac{e^{-\lambda} \lambda^j}{j!} \right] = \int_0^\infty \frac{e^{-\lambda} \lambda^j}{j!} g(\lambda) d\lambda \qquad (1)$$

for $j \in \mathbb{N}$, where the operator $\mathbb{E}_g[\cdot]$ represents the expectation under the mixing distribution g.

5.2 ML estimation on IRM-M

By modifying the model, we have changed the problem of estimating the static parameters $\lambda_1, \lambda_2, \ldots, \lambda_K$, to that of estimating the mixing distribution g.

Problem Statement (IRM-M): *Given the measured request counts* $\{n_1, n_2, \ldots, n_{K_0}\}$*, determine the catalog size* $\hat{K}$ *and the mixing density* $\hat{g}$ *such that an i.i.d. mixed Poisson sample* $\{N_1, N_2, \ldots, N_{\hat{K}}\}$ *is the "closest" to the set* $\{n_1, n_2, \ldots, n_{K_0}, 0, \ldots, 0\}$*, with* $\hat{K} - K_0$ *zeros at the tail.*

We now show how this problem can be solved via a ML method. Let $J = \max_{k=1}^{K_0} \{n_k\}$ be the maximum number of requests over all documents, and let

$$\mu_j = \frac{1}{K_0} \sum_{k=1}^{K_0} \mathbb{1}\{n_k = j\}$$

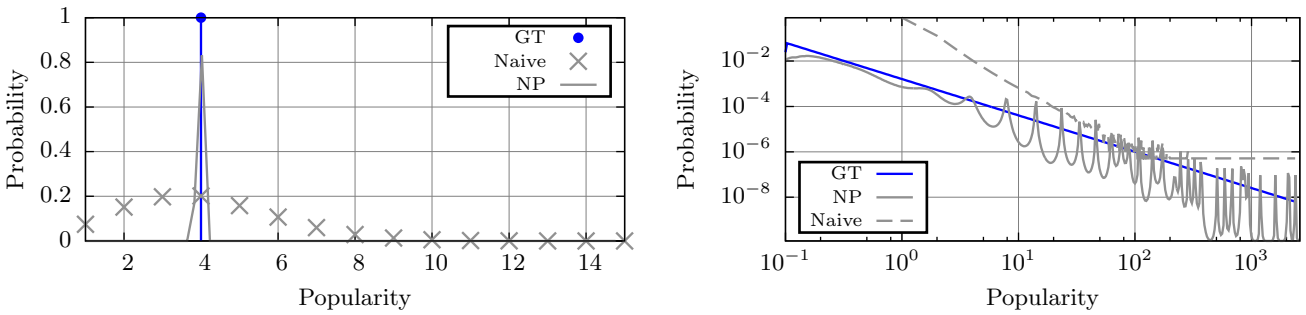

Figure 3: *Mixing distribution obtained via the non-parametric methods for the* `#delta` *(left) and* `#prt` *(right) traces*

be the proportion of documents with j requests, $1 \leq j \leq J$. Using (1), the log-likelihood $\ell(g;\mu)$ of the popularity distribution g for the observations $\mu = (\mu_j)_{j\geq 1}$ reads as

$$\ell(g;\mu) = \sum_{j=1}^{J} \mu_j \log \mathbb{P}[N = j \mid N > 0]$$
$$= \sum_{j=1}^{J} \mu_j \log \mathbb{E}_g\left[\frac{e^{-\lambda}\lambda^j}{j!}\right] - \log \mathbb{E}_g\left[1 - e^{-\lambda}\right].$$

We remark that in this setting, the catalog size K is decoupled from the popularity distribution. Thus, we can first obtain an estimator $\hat{g}$ of the mixing distribution g, and then approximate K by

$$\hat{K}^{\mathrm{ml}} = \frac{K_0}{\mathbb{E}_{\hat{g}}[1 - e^{-\lambda}]} \tag{2}$$

which is asymptotically close to the ML estimator.

We now proceed with the detailed form of the likelihood function for the *parametric* and *non-parametric* estimation procedures. In both approaches, we numerically solve the problems with a generic non-linear optimization solver in MATLAB based on an interior point algorithm. Our code is freely available online.[1] We discuss the use of specialized algorithms in Section 7.

5.2.1 Parametric Estimation

In this setting, we determine the mixing distribution g within a parametric family of density functions. The choice of that parametric family relies on an a-priori knowledge. The computation of the ML estimator obviously depends on this choice, and due to space restriction, we here limit ourselves to the two-parameter Pareto family with densities $g(x) = \alpha x_m^{\alpha}/x^{\alpha+1}$ for $x > x_m$, with α, x_m the shape and scale parameters, respectively. The log-likelihood function $\ell = \ell(\alpha, x_m; \mu)$ then reads

$$\ell = \sum_{j=1}^{J} \mu_j \log \frac{\Gamma(j-\alpha, x_m)}{j!} - \log\left(\alpha x_m^{\alpha} - \Gamma(-\alpha, x_m)\right).$$

5.2.2 Non-Parametric Family

In the absence of a-priori knowledge about the distribution g, the non-parametric (NP) approach provides a method to obtain an estimator. In this setting, we determine a discrete distribution g of the form $\mathbb{P}[\lambda = x_i] = \theta_i$ for $1 < i < I$. The log-likelihood correspondingly reads

$$\ell(\theta;\mu) = \sum_{j=1}^{J} \mu_j \log \sum_{i=1}^{I} \theta_i \frac{e^{-x_i} x_i^j}{j!} - \log \sum_{i=1}^{I} \theta_i (1 - e^{-x_i}).$$

5.3 Hit Ratio Analysis

As detailed in the Appendix 8, the IRM-M model proves to be tractable for evaluating the performance of an LRU cache. In particular, the so-called "Che approximation" is easily adapted to the IRM-M case; furthermore, we are able to derive formulas for the transient analysis of the hit ratio, when starting from an empty cache.

6. NUMERICAL EVALUATION

The accuracy of the parameter estimation can be evaluated at three different levels, as expressed by the following questions: **(1)** Is the estimated popularity density close to the actual popularity density? **(2)** Is the request flow predicted by the model statistically similar to the actual request flow? **(3)** Is the performance indicator of the fitted model, e.g., the hit ratio, accurately predicted?

Throughout this section, we assess the precision of a curve estimate by computing the so-called *mean absolute percentage error* (MAPE). More precisely, the MAPE between a reference sequence of points $(x_i)_{1\leq i\leq N}$ and an estimate sequence $(y_i)_{1\leq i\leq N}$ is defined by

$$\mathrm{MAPE}(X,Y) = \frac{1}{N}\sum_{i=1}^{N} \frac{|y_i - x_i|}{|x_i|}.$$

6.1 Estimation of popularity distribution

First, we start with the most general question, that is, the estimation of the mixing distribution. Such an inverse problem is known to be ill-posed.

For the NP estimation, we obtain an estimate $\hat{g}^{\mathrm{NP}}$ of the popularity density by applying the NP method, using a support with 0.01 as lower bound, exponentially increasing spacings and an upper bound slightly larger than the maximum of observed requests (e.g., 2 400 for `#prt` and 16 for `#delta`). The naive fitting corresponds to the empirical measure of the request counts, that is, the mixture of Dirac measures $\frac{1}{K_0}\sum_{k=1}^{K_0} \delta_{n_k}(.)$.

We observe in Figure 3 the NP estimator of the mixing distribution for the `#delta` and `#prt` datasets. In the `#delta` case, the ground-truth is a Dirac measure at $\lambda = 4$, and the naive method fails at correctly estimating its shape, whereas the ML estimator concentrates its mass around the

[1]Code : `http://www.olmos.cl/code/mixed_poisson.tgz`

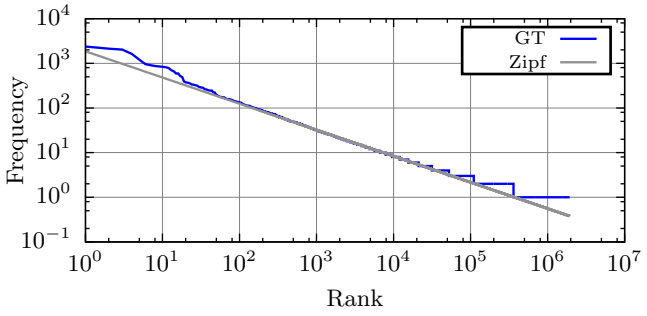

Figure 4: *Rank frequency distribution for the* `#prt` *trace*

value $\lambda = 4$. In the `#prt` case, the estimated distribution is irregular, tending to accumulate mass at certain points (see Section 7.2 for possible regularization solutions). This concentration is no surprise, since in the non-censored case the ML estimator is discrete probability distribution [15]. The peaks, nevertheless, capture the power law trend, as reflected by the good estimation quality of the mixture distribution. In contrast, the naive method fails at correctly estimating both the trend of distribution body and its tail.

Using Equation (2), we also calculate the catalog size, giving $\hat{K} \approx 11\,600\,000$ (resp. 105 278) for the `#prt` (resp. `#delta`) case. This represents a relative error of 11.6% and 5.2%, respectively. Following Equation (2), it shows that estimating the probability that a document receives no request for the duration of the trace, based on the very same trace, is a difficult task. As a consequence, this error is not negligible. It is, however, smaller, and even more significantly in the `#prt` case, than the relative error of the naive method (recall that $\hat{K}^{\text{nv}} = K_0 = 1\,900\,000$ and $\hat{K}^{\text{nv}} = 92\,046$ for the `#prt` and `#delta` traces, respectively).

When some a priori knowledge about the distribution shape is available, the estimates can be improved via the parametric approach. In the `#prt` case, the resulting Pareto fit gives the estimates $\hat{\alpha} = 1.597$ and $\hat{x}_m = 0.099$ that are very close to the original parameters $\alpha = 1.6$ and $x_m = 0.1$. We compare these results to that of the "log-log" approach, which consists in estimating the tail index by fitting a least square approximation to the log-log rank-frequency plot, as shown in Figure 4. The rank frequency plot roughly decays as $1/\alpha$. Using the first 20 000 objects to compute the regression, the estimation gives 1.704, which is worse than the ML estimate.

6.2 Request flow estimation

In this section, we specify the discussion by estimating the zero-censored request count distribution (or mixture distribution in statistical terms) $\mathbb{P}[N = j \mid N > 0]$, $j \geq 1$.

For the naive approach, we generate 50 000 IRM traces using the estimated parameters. We then calculate the average empirical distribution of the request per document. The number of generated traces ensures a coefficient of variation lower that 10^{-4} for all points of the distribution. As regards the ML approach, using the $\hat{g}^{\text{np}}$ density, we compute the associated zero-censored request distribution using (1).

In Figure 5, we show the resulting zero-censored request distribution estimated by each method. For comparison, we include the real mixture distribution for the `#prt` dataset, which can be calculated explicitly. For the `#yt` and `#vod` datasets, we show instead the observed request distribution.

Two issues are raised by the naive approach, that are not present in the maximum likelihood estimation:
– first, at the head of the distribution, where most of the mass is concentrated, large estimation errors are produced by the naive approach. Such errors produce a mass shift towards the tail of the distribution. On the contrary, the NP estimation matches perfectly the head of the distribution;
– second, the naive method over-fits the tail of the distribution. We observe in Figure 5d that the naive estimate shows a "horizontal branch" at the tail, and differs significantly from the ground-truth that is approximately a straight "diagonal" line. This horizontal branch is in fact a few isolated masses, though they look as a line on the figure. The naive estimation therefore concentrates the mass of the ground-truth distribution on a few points. On the other side, the ML estimation correctly estimates the trend of the distribution at all scales, though noise inaccuracies appear at the tail. This is quantified by the MAPE of 1.67 for the ML estimation, whereas the naive method leads to a MAPE of 668, for the full range distribution. As regards the `#yt` and `#vod` cases in Figures 5e and 5f, we similarly observe the same horizontal branch at the tail for the naive distribution. In the absence of available ground-truth, we do not compute the MAPE, but the similarity of behavior hints that the ML method also performs better on these traces.

6.3 Hit Ratio Estimation

We finally compare the hit ratios predicted by the IRM-M model with popularity distributions fitted using the naive and the ML methods, both for the `#prt` and `#yt` traces.

Figure 6 shows the obtained hit ratio curve in each case. The ground-truth curves are obtained by simulation of a LRU cache fed by the shuffled traces. The Naive (resp. NP) curves are obtained when using Formula (6) (resp. (9)) with the parameters obtained by the naive (resp. NP) method. Finally, the Zipf curve, for the `#prt` trace, corresponds to the hit ratio prediction when using the "log-log" parametric fitting method detailed in Section 6.1.

The naive approach leads to small inaccuracy for the `#yt` trace and large errors for the `#prt` trace, with respective MAPE of 0.06 and 1.44. This difference in estimation accuracy can be explained by the variability of the random variable N. Indeed, in the `#yt` dataset, documents receive an average of 7.3 requests per document, whereas this average decreases to 1.4 in the `#prt` trace. It follows that the coefficient of variation of the request count distribution is greater in the `#prt` trace than in the `#yt` trace. As expected, the inaccuracy of the naive method is greater for the former than for the latter. Note also that from an operational point of view, the focus is on the miss ratio, which determines the dimensioning requirements upstream of the cache. The inaccuracy of the naive hit ratio prediction for the `#yt` dataset becomes relatively significant in this context. As shown by the Zipf curve, the knowledge of a relevant parametric family allows us to improve the hit-ratio estimation. The error, however, remains significant with a MAPE of 0.96. In contrast, the non-parametric ML curves match perfectly the original ones, as shown by the MAPE of 0.002 for the `#yt` trace and 0.005 for the `#prt` trace. We conclude that, as regards hit ratio, our estimation method accurately estimates the model parameters. In contrast, in the Zipf case, a seemingly small error of 0.1 in the estimation of the tail exponent

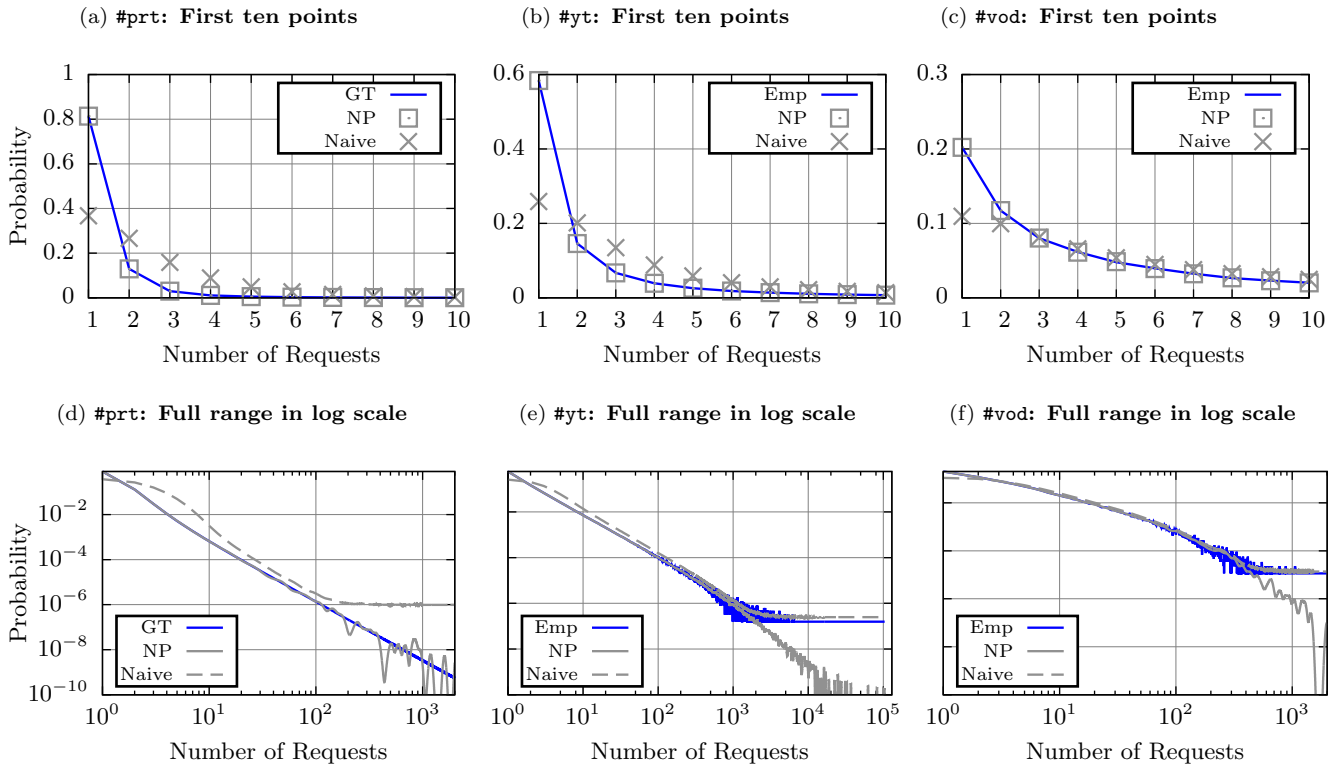

Figure 5: *Censored mixture distribution estimations obtained with the non-parametric method*

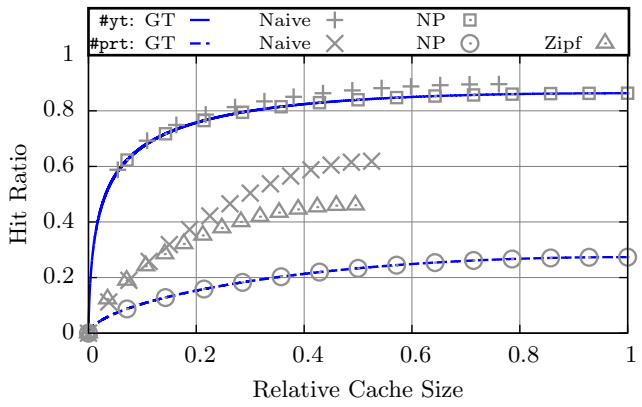

Figure 6: *Hit ratio for* `#yt` *and* `#prt` *datasets. The cache size is normalized with respect to that of the ground-truth in the* `#prt` *case and with respect to* $\hat{K}^{\text{ml}}$ *in the* `#yt` *case.*

leads to a significant error in the hit ratio estimation.

7. DISCUSSION AND CONCLUSION

7.1 Other Applications and Extensions

Since our methodology requires only the statistics about the number of requests per document, the presented estimation method for content popularity can be readily applied in use-cases other than caching performance. For example, the estimations can be used for dimensioning the bandwidth in the access network for VoD or TV multicast services or even predicting the demand for content in marketing studies.

Additionally, the wide applicability of the ML estimators makes our method a viable option for other traffic models. In particular, our framework can be extended to to renewal [6, 1] and cluster processes [17, 21]. In these cases new challenges arises, due the reformulation of the ML method. For example, the randomized parameter in other traffic models is not univariate, but multivariate [17] or even an stochastic process [6]. Another factor to consider is time censure, due the greater impact of the time variable in stochastic models other than IRM-M.

7.2 Maximization techniques

The main current limitation of our maximization approach is that the estimated mixing density exhibits a lot of peaks, which is consistent with the results of Lindsay [15]. This might be a problem when one aims at understanding the nature of the popularity distribution.

A possible solution to enforce smoothness in the mixing density estimation is to introduce a penalization for the irregularities. Classical candidates for such a penalization are the L^2-penalization or a logarithmic penalization $R(\theta) = \sum_{i=1}^{I}(\theta_{i+1} - \theta_i)(\log\theta_{i+1} - \log\theta_i)/(x_{i+1} - x_i)$. One then maximizes $\ell(\theta;\mu) - \rho R(\theta)$, where ρ represents the trade-off factor between fitness and smoothness. Regularization here comes at the price of choosing the right penalization function $R(\cdot)$ and the right value of ρ and in our case, the results have been satisfactory only for concentrated mixing distributions.

Another possibility is to exploit the fact that the peaks conserve the overall trend of the distribution. We thus extract the peak locations. A second ML optimization is then performed using these peak locations as the new support. Though non-standard, this gives satisfactory results for the `#prt` dataset (not shown here due to lack of space).

7.3 Summary of results

In this paper, we have presented and solved the inverse problem that consists in estimating from a trace the popularity parameters for a performance model. A key point in our approach is that we consider the probability that a document receives a given number of requests, rather than the probability that a request is directed to a given document. This representation is consistent with recently developed caching models [17, 21, 6]. Moreover, it allows us to avoid the fitting of a rank-frequency plot, which is in essence an order statistic and exhibits over-fitting. Our second contribution on the modeling aspects is that we consider popularities as random variables, rather than parameters, leading to a mixture model tractable via ML methods. We have illustrated our method in the case of cache performance evaluation but our framework is applicable and extensible to other settings.

The inverse problem stems from the random nature of the requests count N for a given document. In particular, a traffic trace contains a single sample of these requests counts. The accuracy of any method that aims at fitting independently the popularity of each document is therefore limited by the inherent variability of the random variable N. The importance of using a sound methodology correspondingly increases when the variability of the request counts is large, which is typically the case when N is small.

Determining the parameters of the model allows one to use the performance for diverse objectives, including the dimensioning of operational networks or the design of new mechanisms. More importantly, in contrast with simulation-based analysis, it enables one to more easily explore *what-if* scenarios, by keeping some parameters at their current value and modifying others to reflect future or possible changes.

8. APPENDIX

We here detail the derivation of hit ratio formulas for the IRM-M model.

8.1 IRM Model

For comprehension purposes, we first briefly review the "Che approximation" method for the hit ratio estimation in the IRM model (additional details can be found in [7]). Given popularities $\lambda_1, \lambda_2, \ldots, \lambda_K$, let $X^k(t)$ denote the number of different documents, apart from the k-th, requested in a time window $[0, t]$, that is,

$$X^k(t) = \sum_{i=1; i\neq k}^{K} \mathbb{1}\{N_i[0,t] \geq 1\}.$$

Let $T_C^k = \inf\{t > 0 : X^k(t) \geq C\}$ be the exit time to level C for process X^k; T_C^k represents the eviction time for content k in a LRU cache of size C, given that it is not requested during this time period. Now, the core of the "Che approximation" consists in the two following steps:

1. all T_C^k have the same distribution, i.e., $\forall k, T_C^k \stackrel{\mathrm{d}}{=} T_C$;
2. the random variable T_C is well approximated by a constant t_C called the "characteristic time". The time t_C is implicitly defined by the equation

$$\sum_{k=1}^{K} \mathbb{E}[\mathbb{1}\{N_k[0,t_C] \geq 1\}] = \sum_{k=1}^{K} 1 - e^{-\lambda_k t_C} = C. \quad (3)$$

Intuitively, t_C is the time when, on average, C different objects have been requested.

In the stationary case, the hit ratio H can then be derived as follows. Using the PASTA property, the hit ratio of document k for a cache of size C is equal to $1 - e^{\lambda_k t_C}$, and by averaging on all documents, it follows that

$$H \approx \frac{1}{\Lambda} \sum_{k=1}^{K} \lambda_k (1 - e^{-\lambda_k t_C}). \quad (4)$$

In the transient case, we simply assume that $T_C^k \leq W$ (the hit ratio does not increase with T_C^k when $T_C^k > W$). By independence, it can be shown (see Proposition 3, [17]) that the average number of hits for the k-th document in a time window of size W, starting from an empty cache, is $\mathbb{E}\big[h(\lambda_k, T_C^k)\big]$ where the expectation carries on T_C^k and the function $h(\lambda, t)$ is defined by

$$h(\lambda, t) = (\lambda W - 1)(1 - e^{-\lambda t}) + \lambda t e^{-\lambda t}, \quad t < W. \quad (5)$$

In consequence, setting $\Lambda = \sum_{k=1}^{K} \lambda_k$, the transient hit ratio $H(W)$ is given by

$$H(W) = \frac{1}{\Lambda W} \sum_{k=1}^{K} \mathbb{E}\Big[h(\lambda_k, T_C^k)\Big].$$

Applying the "Che approximation", we then obtain the following formula for the hit ratio $H = H(W)$:

$$H \approx \frac{1}{\Lambda} \sum_{k=1}^{K} \lambda_k (1 - e^{-\lambda t_C}) + \frac{1}{\Lambda W} \left(\sum_{k=1}^{K} \lambda_k t_C e^{-\lambda_k t_C} - C \right). \quad (6)$$

The second term of (6) vanishes as $W \to \infty$, leading to equality (4) for the stationary hit ratio.

8.2 IRM-M Model

We now address the IRM-M case. We first show how to derive the hit ratio in this setting; we further prove formally the validity of the "Che approximation" in the case where $C = \delta K$ and K tends to infinity.

• Given the popularities $\lambda_1, \lambda_2, \ldots, \lambda_K$, let us define X^k, T_C^k as in the previous section, and let $\delta = C/K$ be the proportion of stored documents. As the popularities are here an i.i.d. sample, and since X^k and T_C^k are independent of λ_k, the previous quantities do not consequently depend on the document index k. In consequence, this validates the first step of the "Che approximation".

For the second step, define the characteristic time t_δ as

$$t_\delta = r^{-1}(\delta) \quad \text{with} \quad r(t) = \mathbb{E}\Big[1 - e^{-\lambda t}\Big], \quad (7)$$

which is equivalent to dividing both sides of (3) by K. Following the same steps as in the previous section, it is easy to derive the following hit ratio formulas:

$$H \approx \frac{\mathbb{E}\big[\lambda(1 - e^{-\lambda t_\delta})\big]}{\mathbb{E}[\lambda]}, \quad (8)$$

$$H(W) \approx \frac{\mathbb{E}\big[\lambda(1 - e^{-\lambda t_\delta})\big]}{\mathbb{E}[\lambda]} + \frac{\mathbb{E}\big[\lambda t_\delta e^{-\lambda t_\delta}\big] - \delta}{\mathbb{E}[\lambda]\, W}. \quad (9)$$

Equations (8) and (9) are the IRM-M analogs of (4) and (6).

• We show that the second step of the Che approximation is asymptotically exact, that is, the random variable T_C can be replaced by the associated characteristic time t_δ. Consider the case where the cache size scales with the catalog size, that is, δ remains constant, and C and K grow to infinity. Recall that the distribution of T_C is given by

$$\mathbb{P}[T_C > t] = \mathbb{P}\left[\sum_{k=1}^{K} \mathbb{1}\{N_k[0,t] \geq 1\} < C\right]$$

for $t \geq 0$, which can be rewritten as

$$\mathbb{P}[T_{\delta K} > t] = \mathbb{P}\left[\frac{1}{K}\sum_{k=1}^{K} \mathbb{1}\{N_k[0,t] \geq 1\} < \delta\right]. \qquad (10)$$

An application of the law of large numbers shows that

$$\lim_{K\to\infty} \frac{1}{K}\sum_{k=1}^{K} \mathbb{1}\{N_k[0,t] \geq 1\} = r(t)$$

almost surely; using (10), $T_{\delta K}$ thus converges in probability to the constant t_δ, for $\delta \in [0, r(W)]$, with $r(W) = \mathbb{E}[K_0]/K$. By the conditioning argument of Proposition 3 in [17], it can be shown that the expectation of the number of hits $H_C = H_{\delta K}$ satisfies the identity

$$\mathbb{E}[H_{\delta K}] = \mathbb{E}[h(\lambda, T_{\delta K})]\,;$$

applying then the bounded convergence theorem (Section 13.6, [22]) to the latter identity and dividing by the expected number of requests $\mathbb{E}[\lambda]$ leads to formulas (8) and (9), as claimed.

9. REFERENCES

[1] D. S. Berger, P. Gland, S. Singla, and F. Ciucu. Exact analysis of TTL cache networks. *Performance Evaluation*, 2014.

[2] Y. Carlinet, T. D. Huynh, B. Kauffmann, F. Mathieu, L. Noirie, and S. Tixeuil. Four Months in DailyMotion: Dissecting User Video Requests. In *8th International Wireless Communications and Mobile Computing Conference (IWCMC)*. IEEE, 2012.

[3] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon. I tube, you tube, everybody tubes: Analyzing the world's largest user generated content video system. In *7th Conference on Internet measurement (IMC)*. ACM, 2007.

[4] A. Clauset, C. R. Shalizi, and M. E. Newman. Power-law distributions in empirical data. *SIAM Review*, 2009.

[5] F. Comte and V. Genon-Catalot. Adaptive Laguerre density estimation for mixed Poisson models. *Electronic Journal of Statistics*, 2015.

[6] N. C. Fofack, P. Nain, G. Neglia, and D. Towsley. Analysis of TTL-based cache networks. In *6th International Conference on Performance Evaluation Methodologies and Tools*. IEEE, 2012.

[7] C. Fricker, P. Robert, and J. Roberts. A Versatile and Accurate Approximation for LRU Cache Performance. In *24th International Teletraffic Congress (ITC)*. IEEE, 2012.

[8] C. Fricker, P. Robert, J. Roberts, and N. Sbihi. Impact of traffic mix on caching performance in a content-centric network. In *Conference on Computer Communications*. IEEE, 2012.

[9] P. Gill, M. Arlitt, Z. Li, and A. Mahanti. YouTube traffic characterization: A view from the edge. In *7th Conference on Internet measurement)*. ACM, 2007.

[10] W. Gong, Y. Liu, V. Misra, and D. Towsley. On the tails of web file size distributions. In *Annual Allerton Conference on Communication Control and Computing*, 2001.

[11] F. Guillemin, B. Kauffmann, S. Moteau, and A. Simonian. Experimental analysis of caching efficiency for youtube traffic in an ISP network. In *25th International Teletraffic Congress (ITC)*. IEEE, 2013.

[12] L. Guo, E. Tan, S. Chen, Z. Xiao, and X. Zhang. The stretched exponential distribution of internet media access patterns. In *27th Symposium on Principles of distributed computing*. ACM, 2008.

[13] C. Imbrenda, L. Muscariello, and D. Rossi. Analyzing Cacheable Traffic in ISP Access Networks for Micro CDN Applications via Content-centric Networking. In *1st International Conference on Information-Centric Networking*, ICN '14. ACM, 2014.

[14] D. Karlis. A General EM Approach for Maximum Likelihood Estimation in Mixed Poisson Regression Models. *Statistical Modelling*, 2001.

[15] B. G. Lindsay. Mixture Models: Theory, Geometry, and Applications. *Institute for Mathematical Statistics: Hayward, CA*, 1995.

[16] P. Loiseau, P. Gonçalves, S. Girard, F. Forbes, and P. Vicat-Blanc Primet. Maximum Likelihood Estimation of the Flow Size Distribution Tail Index from Sampled Packet Data. In *Performance Evaluation Review*. ACM SIGMETRICS, 2009.

[17] F. Olmos, B. Kauffmann, A. Simonian, and Y. Carlinet. Catalog dynamics: Impact of content publishing and perishing on the performance of a LRU cache. In *26th International Teletraffic Congress (ITC)*. IEEE, 2014.

[18] B. N. Oreshkin, N. Regnard, and P. L'Ecuyer. Rate-based daily arrival process models with application to call centers. Technical report, 2014.

[19] J. Roberts and N. Sbihi. Exploring the memory-bandwidth tradeoff in an information-centric network. In *25th International Teletraffic Congress (ITC)*. IEEE, 2013.

[20] F. Roueff and T. Rydn. Nonparametric estimation of mixing densities for discrete distributions. *The Annals of Statistics*, 2005.

[21] S. Traverso, M. Ahmed, M. Garetto, P. Giaccone, E. Leonardi, and S. Niccolini. Temporal locality in today's content caching: Why it matters and how to model it. *Computer Communication Review*, 2013.

[22] D. Williams. *Probability with Martingales*. Cambridge University Press, 1991.