Maximizing Cache Hit Ratios by Variance Reduction

Daniel S. Berger[a], Sebastian Henningsen[a], Florin Ciucu[b], and Jens B. Schmitt[a]

[a]Distributed Computer Systems (DISCO) Lab, University of Kaiserslautern, Germany

[b]Department of Computer Science, University of Warwick, UK

ABSTRACT

TTL cache models provide an attractive unified approximation framework for caching policies like LRU and FIFO, whose exact analysis is notoriously hard. In this paper we advance the understanding of TTL models by explicitly considering stochastic capacity constraints. We find in particular that reducing the variance of the cache occupancy is instrumental to optimize the cache hit ratio in an online setting. To enforce such a desired low variance, we propose a novel extension of the TTL model by rewarding popular objects with longer TTLs. An attractive feature of the proposed model is that it remains closed under an exact network analysis.

1. INTRODUCTION

The performance analysis of classical cache models such as Least-Recently-Used (LRU) or FIFO is known to be a hard problem [8]. Recent progress on timer-driven eviction models (aka TTL caches) has revealed a class of fast approximation schemes which unify the analysis of LRU [3, 5] FIFO, Random Eviction (and further ones) [8], and mixed caching policies [1] (even in the network case [1, 4, 8]).

In a TTL cache each object simply joins the cache, whereas an associated timer (i.e., the Time-to-Live) determines the object's eviction. TTL models are quite versatile in the sense that they capture the move-to-front behavior of LRU, in particular, by resetting the timer of some object o with each corresponding request. Other caching policies are captured by different resetting behavior [1, 8].

To account for caches of finite capacity, an approximate TTL model abstracts from the capacity-interactions of objects using a single timer – known as the *characteristic time*. Formally, express the number of objects in the cache at time t (the cache occupancy) as the sum of the individual objects' indicator functions $C(t) := \sum_o \mathbb{1}_{o\in\text{Cache}}(t)$. The characteristic time T is derived as the solution of

$$C \approx \mathbb{E}[C(t)] = \sum_o \mathbb{E}[\mathbb{1}_{o\in\text{Cache}}(t)] \ . \tag{1}$$

The indicator functions depend on T, e.g., $\mathbb{E}[\mathbb{1}_{o\in\text{Cache}}(t)] = e^{-\lambda_o T}$ for Poisson arrivals of rate λ_o to each object o. Moreover, a unique solution to (1) is guaranteed by the monotonicity of $\mathbb{E}[\mathbb{1}_{o\in\text{Cache}}(t)]$ in T.

While the approximation in (1) was shown to be relatively accurate in simulations [5, 8], due to a smoothing out behavior in the long run, the underlying TTL cache model frequently underruns or overruns the capacity constraint.

The goal of this paper is to much more rigorously analyze caches of finite capacity by using a stochastic capacity constraint in distribution, rather than in the first moment only, as in (1). Concretely, we consider the problem of optimizing the cache hit ratio in a setting with N objects, each with an arrival rate λ_o (e.g., according to a Zipf popularity law), and a finite cache capacity of C:

$$\text{maximize} \quad H = \sum_{o=1}^{N} \frac{\lambda_o}{\lambda} p_o \tag{2}$$

$$\text{subject to} \quad \mathbb{P}[C(t) \geq C] \leq \varepsilon \ . \tag{3}$$

H is the overall cache hit ratio (for $\lambda = \sum_{o=1}^{N} \lambda_o$), p_o are the individual objects' hit ratios, whereas ε is the violation probability of the enforced stochastic capacity constraint.

The offline version of the optimization problem can be easily solved by assigning timer values, for each object o, proportional to the corresponding arrival rates λ_o. In turn, the online version is particularly hard because the caching policy is unaware of the λ_o's. To solve it, we retain the idea of using a single timer value T for all objects, as in (1); note that T is the only optimization variable in (2)-(3).

The key observation is that the cache occupancy $C(t)$ should have both a high expectation and a low variance. The first property is necessary in general for optimizing the overall hit ratio. In turn, the second property does not suffice to guarantee large hit ratios, since the cache may be dominated by unpopular objects. To actually enforce that large hit ratios correspond to the most popular objects, Section 2 proposes a novel TTL caching policy which is able to adapt to the objects' popularities.

Before we give the new policy, together with its analysis, we need to make the stochastic constraint from (3) explicit, and yet incorporate the previous key observation. We do so by invoking Bernstein's inequality [2], i.e.,

$$\mathbb{P}[C(t) > C] \leq exp\left\{ \frac{-\left(C - \mathbb{E}[C(t)]\right)^2}{\text{Var}[C(t)] + \left(C - \mathbb{E}[C(t)]\right)/3} \right\} \ . \tag{4}$$

Note that this concentration inequality (unlike others, e.g., Hoeffding's inequality) captures the desired property that the violation probability of the stochastic capacity constraint decays with smaller $Var[C(t)]$.

2. A TTL POLICY WITH LOW VARIANCE

For the sake of tractability this section assumes that the objects o are requested according to independent Poisson

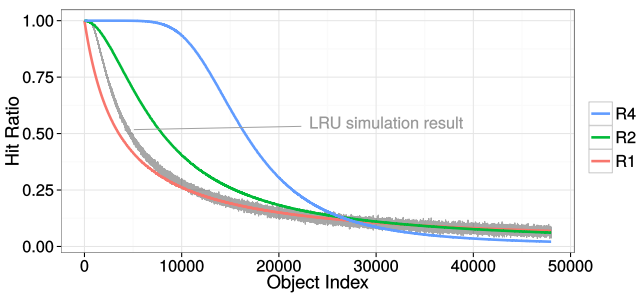


Figure 1: The analytical hit ratio p_o for the classical $\mathcal{R}_1$ model and the new $\mathcal{R}_2$ and $\mathcal{R}_4$ policies (objects ordered by popularity for a subset out of $1e6$ objects), and empirical hit ratios from LRU simulations.

processes with rates λ_o (cf. the Independent Reference Model (IRM) [3, 5, 8]). Then $\mathbb{E}[\mathbb{1}_{o\in\text{Cache}}(t)]$ is the stationary hit ratio p_o, whereas the variance of $C(t)$ simplifies to

$$\text{Var}[C(t)] = \sum_{o=1}^{N} \text{Var}[\mathbb{1}_{o\in\text{Cache}}(t)] = \sum_{o=1}^{N} p_o\,(1-p_o)\ .$$

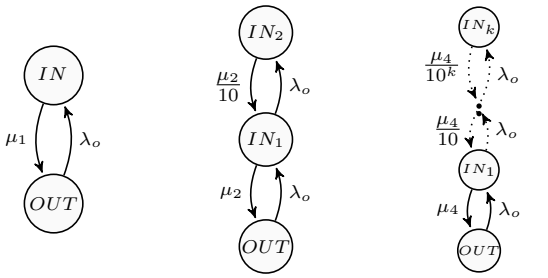
The variance is thus minimized when the hit ratios p_o's are either 0 or 1. If the object indices were known (and ordered by popularity), then a zero variance would be guaranteed by $p_o = 1$ for $o \leq C$ and $p_o = 0$ for $o > C$ (i.e., the Least-Frequently-Used (LFU) policy). In contrast, practical cache policies have a high variance because the cache hit ratio slowly decays with the object popularity. For example, Figure 1 shows this behavior for the hit ratios from LRU simulations and for a TTL model called the $\mathcal{R}$ model [1], whereby timers are reset at every request arrival (and which has been used to approximate LRU[1]).

To enforce a much sharper decay of hit ratios, and thus decrease the cache (occupancy) variance, we propose a stage-based version of the $\mathcal{R}$ policy. Instead of equally treating objects, our key idea is to "reward" popular objects with longer timers. Because a cache has no knowledge of the actual popularities (and these might change over time), we split the cache into several stages and move objects with each hit "forward" into a stage with a longer timer. Conversely, each time the timer expires the object is moved "backwards", until it reaches the first stage. When the timer in the first stage expires, the object is evicted from the cache. Newly admitted objects start in the first stage.

If there are k stages, we call this the $\mathcal{R}_k$ TTL cache policy. The special case $\mathcal{R}_1$ recovers the $\mathcal{R}$ model, and for $k \to \infty$ the model approaches a behavior similar to LFU. Interestingly, however, even small numbers of stages turn out to be quite effective. Subsequently, we compare $\mathcal{R}_1$ to $\mathcal{R}_2$ and $\mathcal{R}_4$ configurations.

For Poisson arrivals, the new cache policy can be analyzed for a tagged object o with a simple Markov chain that represents whether o is OUT of the cache or IN the cache, and in which stage it is. For simplicity, we represent the timers as exponential random variables with rate $\mu = 1/T$ – although quasi-deterministic timers are possible using Proposition 3. Note that the timers are independent of the object index o

[1]The original approximation uses deterministic T, we use exponentially distributed T for modeling purposes. This is why the difference between $\mathcal{R}$ and the LRU simulations seems rather large.



(a) $\mathcal{R}_1$ model (b) $\mathcal{R}_2$ model (c) $\mathcal{R}_k$ model

Figure 2: The cache state of a tagged object o for a classical LRU model ($\mathcal{R}_1$) and the new $\mathcal{R}_k$ policies. Note that μ_k are uniform over all objects but specific for the policy's number of stages k.

(as in [3, 5, 8]), but depend on k (i.e., we write μ_k if there are k stages). Figure 2 illustrates the Markov chain model for the case when each stage's timer is ten times the previous stage's timer in the mean.

We obtain the steady-state hit ratio for o as the sum of the limiting probabilities of a model's IN-states, e.g.,

$$p_o^{\mathcal{R}_1} = \frac{\lambda}{\lambda+\mu_1} \qquad \text{and} \qquad p_o^{\mathcal{R}_2} = \frac{\mu_2\lambda + 10\lambda^2}{\mu_2^2 + \mu_2\lambda + 10\lambda^2}\ .$$

As an example, consider an object universe of $N = 1e6$ objects under a Zipf popularity law. To numerically illustrate the reduction in variance, we compare the policies for the same expected cache size, i.e., $\mathbb{E}[C(t)] = 2e4$. Figure 1 shows the resulting hit ratios. As expected, popular objects stay longer in $\mathcal{R}_2$ and $\mathcal{R}_4$ than in the $\mathcal{R}_1$ cache. Additionally, the hit ratio decays much faster for unpopular objects (in particular for $\mathcal{R}_4$). These two effects result in a variance that is $\approx 18\%$ smaller for $\mathcal{R}_2$ and $\approx 65\%$ smaller for $\mathcal{R}_4$.

This reduction in variance can be generalized to the $\mathcal{R}_k$ case by bounding each p_o away from $1/2$ with increasing k. We summarize this result in the following Proposition.

PROPOSITION 1. *Under the IRM and a power law popularity model,* $\text{Var}[C_{\mathcal{R}_k}(t)]$ *decreases with* k *when* $\mathbb{E}[C(t)]$ *is held constant.*

We conclude this section by demonstrating that the lower variance actually translates into higher overall cache hit ratios H. In order to do this, we calculate the maximal hit ratio for each policy under the violation probability ε, i.e., we solve the optimization problem using the Bernstein inequality. Figure 3 shows a plot of the resulting hit ratio over ε for $C = 2e4$. The new caching policies outperform the classical $\mathcal{R}_1$ model by $\approx 8\%$ for $\mathcal{R}_2$ and $\approx 15\%$ for $\mathcal{R}_4$ (for any ε). This concludes the description of the $\mathcal{R}_k$ model.

3. ANALYSIS OF CACHING NETWORKS

This section addresses the practical case of caching networks [4, 8, 1]. The technical challenge in the analysis of caching networks is that the output of a cache (the miss process) is very different from a Poisson process [4, 1]. The request streams' stochastic properties are further complicated by network operations like merging and splitting. Nevertheless, we find that networks of $\mathcal{R}_k$ caching policies can be exactly analyzed. We achieve this by showing that the class of Markov arrival processes (MAPs) is closed under the $\mathcal{R}_k$ caching operation. Because MAPs are closed under merging

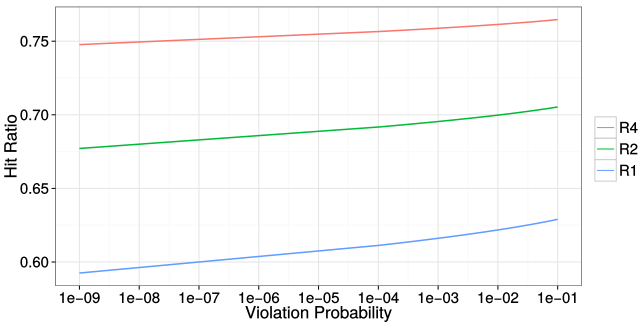


Figure 3: Plotting the analytical hit ratio over the maximal violation probability ε (on log scale) shows significant gains of the new TTL caching policies. These bounds are only slightly pessimistic: the approximation (1) gives 0.64, 0.72, and 0.77, as the hit ratios for $\mathcal{R}_1$, $\mathcal{R}_2$, and $\mathcal{R}_4$, respectively.

and splitting, we can analyze caching networks similar to the recent exact analysis of classical TTL policies [1].

We start by using a phase-type (PH) distribution distribution to describe the cache eviction behavior.

DEFINITION 2. *We call P an* eviction distribution*, if P is a $(k \times l)$-phase PH distribution that is organized in k stages of each l states and is absorbed in state 0. P is further characterized by the probability vectors $\vec{a}_i = a_{i,1}, \ldots, a_{i,l}$ which give the probability of starting in each state of stage i.*

As this is a generalization of the k exponential stage model from Figure 2(d), we state how to formulate this model using an eviction distribution as an example:

$$P = \begin{pmatrix} 0 & 0 & 0 & \ldots & 0 \\ \mu & -\mu & 0 & \ldots & 0 \\ 0 & \mu/10 & -\mu/10 & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \mu/10^k & -\mu/10^k \end{pmatrix} \text{ and } \vec{a}_i = 1.$$

Note that the eviction distribution model is quite general, e.g., it can also capture stage holding distributions with low coefficient of variance to further reduce the overall randomness. We next state the main result that characterizes the output process for these generalized $\mathcal{R}_k$ caches.

PROPOSITION 3. *Consider an $\mathcal{R}_k$-policy characterized by the eviction distribution P and arriving requests represented by the m-state MAP $M = (D_0, D_1)$. Further assume that P and M are independent. Then, the output process $M' = (D_0', D_1')$ is a MAP and defined by*

$$\mathbf{D}_0' = (\mathbf{P} \oplus \mathbf{D}_0) + \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \vec{a}_2 \otimes \mathbf{D}_1 & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \vec{a}_3 \otimes \mathbf{D}_1 & \mathbf{0}.. & \mathbf{0} \\ \vdots & \ldots & \vdots & \vdots & \ddots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} & \vec{a}_{k-1} \otimes \mathbf{D}_1 \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} & \vec{a}_k \otimes \mathbf{D}_1 \end{pmatrix}$$

$$\mathbf{D}_1' = \begin{pmatrix} \mathbf{0} & \vec{a}_1 \otimes \mathbf{D}_1 & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} \\ \vdots & \vdots & \vdots & & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} \end{pmatrix}$$

where $\mathbf{0}$ are 0-matrices of size $m \times m\,l$, and $\oplus$ and $\otimes$ denote the Kronecker Plus and Kronecker Product, respectively.

This result enables the exact analysis of $\mathcal{R}_k$ cache networks, even for the heterogeneous case with the caching policies analyzed in [1]. Note, however, that due to the fast growing number of states (the number of states is $m \times k \times l$) this exact analysis is only practical for medium-sized cache networks.

4. DISCUSSION AND CONCLUSIONS

In this paper we have proposed a new class of stage-based TTL cache models with low variance, and have shown how to analyze them under a stochastic capacity constraint. A question we have not answered concerns the value of the optimal k. While the formulation of the optimization problem from (2) suggests $k \to \infty$, this is clearly not practical because the popularity of individual objects may change over time (although the overall popularity law may be considered invariant). To include this additional constraint into the optimization formulation, one may consider novel ways for modeling popularities. One approach would be the shot-noise popularity model which has recently been shown to be compatible with TTL caching analysis [7].

Our numerical evaluations have focused on small values of k (2 and 4), which were sufficient to significantly reduce the variance of $C(t)$, and thus boost the hit ratio under the capacity constraint. Interestingly, our finding on the efficiency of $\mathcal{R}_4$ parallels the findings of a recent measurement paper [6]. The authors report that replacing LRU with S4LRU – which works similarly to $\mathcal{R}_4$ – leads to a considerable improvement of the hit ratio. In this regard, our $\mathcal{R}_k$ model may be considered as the first analytical approximation for the class of "SkLRU" caching policies.

As a final remark, we point out that the hit ratios from our analysis (cf. Figure 3) differ only between 6% (for $\mathcal{R}_1$) and 2% (for $\mathcal{R}_4$) to the approximation (1), even for small violation probabilities. This validates the idea of using stochastic capacity bounds and shows that replacing TTL approximations with performance bounds costs little accuracy.

5. REFERENCES

[1] D. S. Berger, P. Gland, S. Singla, and F. Ciucu. Exact analysis of TTL cache networks. *Performance Evaluation*, 79(0):2 – 23, 2014.

[2] S. Bernstein. The theory of probabilities, 1946.

[3] H. Che, Y. Tung, and Z. Wang. Hierarchical web caching systems: Modeling, design and experimental results. *IEEE JSAC*, 20(7):1305–1314, 2002.

[4] N. Choungmo Fofack, M. Dehghan, D. Towsley, M. Badov, and D. Goeckel. On the performance of general cache networks. In *Proc. of VALUETOOLS*, 2014.

[5] C. Fricker, P. Robert, and J. Roberts. A versatile and accurate approximation for lru cache performance. In *Proc. of the ITC*, page 8. ITC, 2012.

[6] Q. Huang, K. Birman, R. van Renesse, W. Lloyd, S. Kumar, and H. C. Li. An analysis of facebook photo caching. In *Proc. of ACM SOSP*, pages 167–181. ACM, 2013.

[7] E. Leonardi and G. L. Torrisi. Least recently used caches under the shot noise mode. *preprint arXiv:1411.4759*, 2014.

[8] V. Martina, M. Garetto, and E. Leonardi. A unified approach to the performance analysis of caching systems. In *Proc. of IEEE INFOCOM*, pages 2040–2048, 2014.