Soft-TTL: Time-Varying Fractional Caching

Jasper Goseling and Osvaldo Simeone

***Abstract*—Standard time-to-live (TTL) cache management prescribes the storage of entire files, or possibly fractions thereof, for a given amount of time after a request. As a generalization of this approach, this letter proposes the storage of a time-varying, diminishing, fraction of a requested file. Accordingly, the cache progressively evicts parts of the file over an interval of time following a request. The strategy, which is referred to as soft-TTL, is justified by the fact that traffic traces are often characterized by arrival processes that display a decreasing, but non-negligible, probability of observing a request as the time elapsed since the last request increases. An optimization-based analysis of soft-TTL is presented, demonstrating the important role played by the hazard function of the inter-arrival request process, which measures the likelihood of observing a request as a function of the time since the most recent request.**

***Index Terms*—Caching, TTL, content delivery networks, hazard rate.**

I. Introduction

CACHING of popular files is a key enabling technology for content delivery networks and is finding applications also at the wireless edge (see [1], [2]). Conventionally, an entire file is stored in a cache when a previously uncached content is requested by some user. Furthermore, an increasingly popular approach for managing the cache memory prescribes that a file be retained in the cache for a fixed amount of time, known as Time-to-Live (TTL), which is generally to be optimized based on the statistics of the users' requests [3]. By properly adjusting the parameters in a TTL cache, it has been shown that TTL caching can mimic the behavior of traditional caching mechanisms such as Least Recently Used (LRU) and Least Frequently Used (LFU) [4].

In contrast to the classical design of caching entire files, it is well understood that there are potential advantages to be accrued by storing only fractions of popular files: (*i*) Users may only consume part of the content, e.g., watch only the first few minutes of a movie [1], [5], [6]; and (*ii*) In streaming applications, the cached fraction can be used to fill the buffers and immediately start playback, while the rest of the file is downloaded from the library [2], [7]. Under these conditions, caching a fraction of a file helps freeing up space to store more files, hence sharing the benefits of caching across a larger range of popular contents.

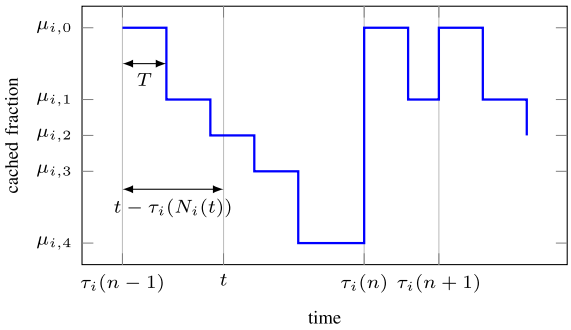

Fig. 1. Soft-TTL via time-varying fractional caching: A file i requested a times $\tau_i(n)$ for $n = 1, 2, \ldots$, and a decreasing fraction of the file is stored in the cache for a subsequent interval of time.

In this letter, we generalize the idea of caching fractions of files by enabling the storage of a time-varying, diminishing, fraction of a requested file. This is illustrated in Fig. 1, in which, upon receiving a request at times $\tau_i(n)$ for $n = 1, 2, \ldots$, a fraction of a previously uncached file i is stored, and the cache progressively evicts parts of the file over a subsequent interval of time. The approach, which we refer to as *soft-TTL*, can be seen as a generalization of the concept of TTL, whereby an entire file is cached for a given amount of time after a request, as well as of *fractional-TTL*, in which a fraction of a file is stored over some time interval following a request.

Soft-TTL is justified by the fact that traffic traces are often characterized by arrival processes that display a decreasing, but non-negligible, probability of observing a request as the time elapsed since the last request increases [8], [9]. Extending the argument in favor of fractional caching, soft-TTL can potentially use the available cache memory more effectively by adapting to the traffic statistics even in the presence of late arrivals.

In this letter, we present an optimization-based analysis of soft-TTL by comparing it on an equal footing with conventional TTL and fractional TTL. We provide analytical results regarding the optimization of the mentioned caching policies given the traffic statistics. We specifically demonstrate the important role played by the *hazard function* of the inter-arrival request process, which measures the likelihood of observing a request as a function of the time elapsed since the last request [10, Ch. 2]. Numerical results show the advantages offered by the additional design degrees of freedom enabled by soft-TTL.

Manuscript received July 16, 2018; accepted September 11, 2018. Date of publication November 23, 2018; date of current version February 22, 2019. The work of O. Simeone was supported in part by the European Research Council through the European Union Horizon 2020 Research and Innovation Program under Grant 725731, and in part by the U.S. NSF under Grant CCF-1525629. The associate editor coordinating the review of this paper and approving it for publication was A. Ksentini. *(Corresponding author: Jasper Goseling.)*

J. Goseling is with the Department of Applied Mathematics, University of Twente, 7500 AE Enschede, The Netherlands (e-mail: j.goseling@utwente.nl).

O. Simeone is with the Centre for Telecommunications Research, Department of Informatics, King's College London, London WC2R 1ES, U.K. (e-mail: osvaldo.simeone@kcl.ac.uk).

Digital Object Identifier 10.1109/LNET.2018.2883245

II. Model

We consider a standard set-up in which a collection of J popular files in a given content library are requested according to independent renewal processes. Each file $i \in \{1, \ldots, J\}$ has size s_i in bits. As illustrated in Fig. 1, let $\tau_i(n)$ for $n = 1, 2, \ldots$ and $i \in \{1, \ldots, J\}$ denote the time of the n-th request for file i; and $N_i(t)$, for $t \geq 0$, the number of corresponding requests up to time t. For notational convenience and without loss of generality, we assume that $\tau_i(0) = 0$. The time between two requests for the same file i has a cumulative distribution

$$F_i(t) = \Pr(\tau_i(n) - \tau_i(n-1) \leq t). \tag{1}$$

We assume that function $F_i(t)$ is continuous, and use $\lambda_i = \mathbb{E}[X_i]^{-1}$ to denote the request rate, where X_i is a random variable distributed according to $F_i(t)$.

We consider a single cache that can store entire files or a fraction thereof. According to the principle of soft-TTL introduced in the previous section, the fraction of a file that is cached can change over time. Let $\mu_i(t)$ denote the fraction of file i that is cached t seconds after its last request, i.e., after the $N_i(t)$th request. Function $\mu_i : [0, \infty) \to [0, 1]$ is non-increasing over time and is referred to as the *caching policy* for file i. The constraint that the caching policy is non-increasing follows from the fact that a cache can evict part of a file from its storage, but it cannot store more than it currently has until a request for the file occurs.

As seen in Fig. 1, we specifically restrict our attention to caching policies that take at most a finite number of values. Furthermore, changes can only occur at equally spaced intervals of time equal to T for an overall amount of time equal to KT for some integer K. Accordingly, we consider caching policies $\mu_i(t)$ that are given as

$$\mu_i(t) = \begin{cases} \mu_{i,0}, & \text{if } t < T, \\ \mu_{i,k}, & \text{if } kT \leq t < (k+1)T, \text{ for } k = 1, \ldots, K-1 \\ \mu_{i,K}, & \text{if } t \geq KT \end{cases} \tag{2}$$

for some cached fractions $1 \geq \mu_{i,0} \geq \mu_{i,1} \geq \cdots \geq \mu_{i,K} \geq 0$.

Broadly speaking, the aim of caching is to minimize the cost of serving requested files. In this letter, following a standard approach (see [3]), we will formulate this task as the maximization of a utility function. To this end, for any file i, we define a utility function $w_i(\mu)$ that measures the utility accrued if file i a requested at a time in which it is cached for a fraction μ. We assume that the functions $w_i : [0, 1] \to [0, \infty)$ are strictly concave and increasing. This choice reflects the scenarios listed in Section I in which caching yields decreasing gains as a larger fraction of a file is stored. The long-term average utility for a given file i is hence given as the limit

$$W_i = \lim_{t \to \infty} \frac{1}{t} \sum_{n=1}^{N_i(t)} w_i(\mu_i(\tau_i(n) - \tau_i(n-1))). \tag{3}$$

We aim at maximizing the standard α-fair utility metric over all files [11], which is defined as

$$\mathcal{W} = (1-\alpha)^{-1} \sum_{i=1}^{J} W_i^{1-\alpha}, \tag{4}$$

where $\alpha \neq 1$ is a fairness parameter. Note that, at the two extremes, setting $\alpha = 0$ corresponds to maximizing the sum utility, while $\alpha \to \infty$ corresponds to the strictest criterion of max-min fairness.

The utility (4) will be optimized over the caching policies $\mu_i(t)$ under a long-term average cache capacity constraint, also considered in, e.g., [3], which is given as

$$\sum_{i=1}^{J} C_i \leq C, \text{ with } C_i = \lim_{t \to \infty} \frac{s_i}{t} \int_0^t \mu_i(s) ds, \tag{5}$$

where C_i is the long-term average cache occupancy of file i. As discussed in the next section, the general problem formulation described here subsumes as special cases the optimization of the conventional TTL and fractional-TTL methods.

To conclude this section, we define the following two key quantities

$$F_{i,k} = F_i((k+1)T) - F_i(kT), \tag{6}$$

$$A_{i,k} = \int_{kT}^{(k+1)T} (1 - F_i(t)) dt, \tag{7}$$

for $k = 0, \ldots, K-1$, as well as $F_{i,K} = 1 - F(KT)$ and $A_{i,K} = \int_{kT}^{\infty} (1 - F(t)) dt$.

III. Preliminaries: TTL and Fractional-TTL

In this section, we formulate optimization models that provide the utility-maximizing TTL and fractional-TTL policies. A fractional-TTL policy is obtained as a special case of the general strategy (2) by imposing that the same fraction ν_i be cached for some TTL period LT defined by an integer $0 \leq L \leq K$, i.e., by setting $\mu_{i,0} = \mu_{i,1} = \cdots = \mu_{i,L} = \nu_j$ and $\mu_{i,L+1} = \mu_{i,L+2} = \cdots = \mu_{i,K} = 0$. The TTL policy further imposes the constraint that the cached fraction be equal to one, i.e., $\nu_i = 1$.

To proceed, we first establish the following general result on the utility W_i and cache occupancy C_i for each file i.

Lemma 1: For $i = 1, \ldots, J$ we have

$$W_i = \lambda_i \sum_{k=0}^{K} w_i(\mu_{i,k}) F_{i,k}, \quad C_i = \lambda_i s_i \sum_{k=0}^{K} \mu_{i,k} A_{i,k}, \tag{8}$$

with probability 1.

Proof: For the utility (3), we can write $\lambda_i \mathbb{E}[w_i(\mu_i(X_i))]$, where the equality holds with probability 1 by the fundamental theorem in renewal reward theory (see [12, Th. 3.6.1]). This is seen by identifying $w_i(u_i(X_i))$ as the reward that is obtained if a request, i.e., a renewal, is observed after time X_i from the previous. A similar argument applies for the cache occupancy C_i in (5). To this end, we interpret the quantity $\int_0^{X_i} \mu_i(s) ds$ as the reward obtained at the end of a renewal cycle. ■

To formulate the optimization of utility (4) under the cache capacity constraints in (5) for fractional TTL we introduce a binary variable $\theta_{i,k}$ such that

$$\theta_{i,k} = \begin{cases} 1, & \text{if } k \leq L, \\ 0, & \text{if } k > L. \end{cases} \tag{9}$$

The resulting optimization problem follows directly from Lemma 1 and is given as

$$\text{maximize} \quad (1-\alpha)^{-1} \sum_{i=1}^{J} \left(\lambda_i \sum_{k=0}^{K} w_i(\mu_{i,k}) F_{i,k} \right)^{1-\alpha}, \quad (10)$$

$$\text{subject to} \quad \sum_{i=1}^{J} \lambda_i s_i \sum_{k=0}^{K} \mu_{i,k} A_{i,k} \leq C, \quad (11)$$

$$1 \geq \mu_{i,0} \geq \mu_{i,1} \geq \cdots \geq \mu_{i,K} \geq 0, \quad (12)$$

$$0 \leq \nu_i \leq 1, \quad (13)$$

$$\theta_{i,k} \in \{0,1\}, \quad (14)$$

$$-\theta_{i,k} \leq \mu_{i,k} \leq \theta_{i,k}, \quad (15)$$

$$\theta_{i,k} - 1 \leq \mu_{i,k} - \nu_i \leq 1 - \theta_{i,k}, \quad (16)$$

where the optimization is over the variables $\{\mu_{i,k}, \theta_{i,k}, \nu_i\}$. The problem above is a mixed-binary convex program that can be solved by modern optimization software such as Gurobi [13]. The corresponding problem for TTL is obtained by setting $\nu_i = 1$.

IV. Soft-TTL

In this section, we tackle the general problem of optimizing the utility (4) under cache capacity constraints (5) for soft-TTL policies. We first demonstrate in Theorem 1 that the problem results in a convex optimization program. We then offer in Theorem 2 an interpretation of the optimal soft-TTL policy in terms of the hazard function of the request processes.

Theorem 1: The problem of optimizing the utility (4) under cache capacity constraints (5) for soft-TTL can be written as the following convex program

$$\text{maximize} \quad (1-\alpha)^{-1} \sum_{i=1}^{J} \left(\lambda_i \sum_{k=0}^{K} w_i(\mu_{i,k}) F_{i,k} \right)^{1-\alpha}, \quad (17)$$

$$\text{subject to} \quad \sum_{i=1}^{J} \lambda_i s_i \sum_{k=0}^{K} \mu_{i,k} A_{i,k} \leq C, \quad (18)$$

$$1 \geq \mu_{i,0} \geq \mu_{i,1} \geq \cdots \geq \mu_{i,K} \geq 0. \quad (19)$$

Proof: The theorem follows directly from Lemma 1 and basic properties of convex functions [14]. ∎

Given the convexity of the problem (17), an optimal soft-TTL policy can be found with any general-purpose convex solver. To obtain further insights into the optimal soft-TTL policy and its relationship to fractional-TTL and TTL, we now consider the special case of the problem (17) with $J = 1$ file. With this simplification, the optimization can be handled in a manner similar to continuous nonlinear resource allocation problems as considered in [15].

To elaborate on this point, for the rest of this section, we drop the index i since we consider $J = 1$ file, and we set $\lambda_1 = s_1 = 1$ without loss of generality. The following theorem details the optimal soft-TTL solution. An interpretation of the result is discussed after the theorem. We note that the extension to multiple files would follow by properly optimizing the partition of the cache across the different files, which generally requires numerical optimization.

Theorem 2: Assume that $J = 1$ and that the utility function w is differentiable, strictly concave, and increasing. For any $\gamma \geq 0$, let $\bar{\mu}_k(\gamma)$ denote the solution[1] of the equality

$$w'(\bar{\mu}_k(\gamma)) = \gamma \frac{A_k}{F_k} \quad (20)$$

and let

$$\mu_k^*(\gamma) = \begin{cases} \mu_{k-1}^*(\gamma), & \text{if } \bar{\mu}_k(\gamma) \geq \mu_{k-1}^*(\gamma), \\ \bar{\mu}_k(\gamma), & \text{if } 0 < \bar{\mu}_k(\gamma) < \mu_{k-1}^*(\gamma), \\ 0, & \text{if } \bar{\mu}_k(\gamma) \leq 0, \end{cases} \quad (21)$$

for $k = 0, \ldots, K$, with $\mu_{-1}^*(\gamma) = 1$. Then, the optimal solution of problem (17) is obtained as $\mu_k^*(\gamma^*)$, for $k = 1, \ldots, K$, where γ^* is the solution of the equality

$$\sum_{k=0}^{K} \mu_k^*(\gamma) A_k = C. \quad (22)$$

Proof: Since the utility function w is increasing and we have $F_k \geq 0$ and $A_k \geq 0$, we can rewrite the optimization problem of Theorem 1 as

$$\min \quad -\sum_{k=0}^{K} w(\mu_k) F_k, \quad (23)$$

$$\text{subject to} \quad \sum_{k=0}^{K} \mu_k A_k = C, \quad (24)$$

$$\mu_k - \mu_{k-1} \leq 0, \quad k = 0, \ldots, K, \quad (25)$$

$$-\mu_k \leq 0, \quad k = 0, \ldots, K, \quad (26)$$

with $\mu_{-1} = 1$. In addition we introduce dual variables γ, ϕ_k and θ_k for constraints (24), (25) and (26), respectively.

We observe that, for any value of the dual variable γ, we can construct an optimal solution by starting from $k = 0$ and by considering $k = 1, 2, \ldots$ consecutively. In fact, for each k, the structure of the problem equals that of [15, Sec. 1] and values for μ_k, ϕ_k and θ_k follow accordingly. Furthermore, since there is a single capacity constraint (24), also the value of γ can be obtained as in [15]. ∎

As detailed in Theorem 2, the optimal soft-TTL policy satisfies the condition

$$w'(\mu_k^{\text{soft}}) \propto \frac{A_k}{F_k} \approx \frac{1}{h(kT)}, \quad (27)$$

where $h(t) = F'(t)/(1 - F(t))$ is the hazard function of the inter-arrival request process, with the prime notation denoting a derivative. The approximate equality in (27) follows by considering the continuous limit $T \to 0$. The hazard function $h(t)$ measures the probability of observing the next request after t seconds since the most recent request. Using (27), and recalling that the derivative w' of the concave utility function is non-increasing, we can draw the following conclusions under the typical assumption of a non-increasing hazard function.

First, if the hazard function decreases very quickly, and hence we have clustered or bursty arrivals, then only the fraction μ_1 is non-zero and TTL is optimal. At the other

[1]For completeness, if $\gamma \frac{A_k}{F_k}$ is above (or below) the range of values for w', then the quantity $\bar{\mu}_k(\gamma)$ should be set to 0 (or 1).

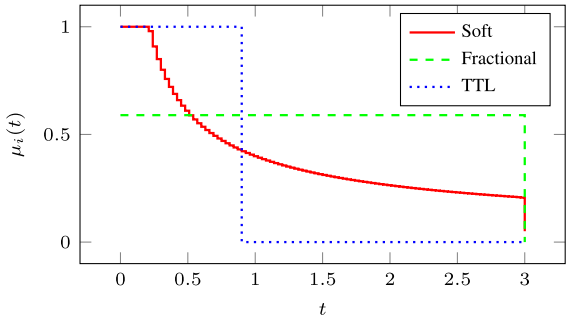

Fig. 2. Optimal policy $\mu_i(t)$ for the TTL, fractional-TTL, and soft-TTL.

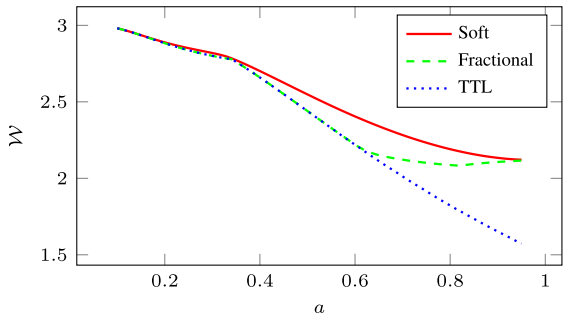

Fig. 3. Optimal utility as a function of Weibull shape parameter a.

extreme, if the hazard function is constant and hence we have a memoryless arrival process, it is generally optimal to cache a constant fraction of the file and fractional TTL is optimal. In the general intermediate case, soft-TTL is required to obtain the optimal performance.

V. Numerical Results

In this section, we present numerical results assuming Weibull distributed inter-arrival times X_i (see [10]). The Weibull distribution has a shape parameter a_i and a scale parameter b_i, where the latter can be found as a function of the arrival rate as $b_i = (\lambda_i \Gamma(1 + a_i^{-1}))^{-1}$, where $\Gamma(\cdot)$ is the Gamma function. The shape parameter determines the hazard function $h_i(t) = b_i^{-a_i} a_i t^{a_i - 1}$, which is a decreasing function of t for $a_i < 1$ and is constant for $a_i = 1$. Values of a_i larger than 1 are not considered, since they yield increasing hazard functions. We consider the same strictly concave utility function $w_i(\mu) = \sqrt{\mu}$ and $a_i = a$ for all files. Unless stated otherwise, we set $J = 3$ files; the size of all files to be normalized to $s_i = 1$; the arrival rates to be equal to $\lambda_i = 1$; the sum-utility obtained with $\alpha = 0$; and the cache capacity constraint to be set to $C = 1.5$. Furthermore, caching policies (2) are optimized with the parameters $K = 100$ and $T = 0.03$.

We first illustrate the optimal policies for TTL, fractional-TTL, and soft-TTL when setting $a = 0.7$ in Fig. 2. This value of a corresponds to a decreasing hazard function with a non-negligible tail. The solution is shown for one of the files given that, by symmetry, the policies for all files were seen to be the equal. The optimal soft-TTL solution is seen to differ from both TTL and fractional-TTL, and it prescribes a decreasing fraction of cached files over time.

In Fig. 3, we show the sum-utility of the three schemes as a function of the shape parameter a. Confirming the insights from the analysis in the previous section, we observe that, when a is very small, and hence the arrivals are clustered, or bursty, TTL is optimal. At the opposite end, when $a = 1$ and the inter-arrival processes are memoryless, fractional TTL is optimal and it offers a significant gain over TTL. Finally, for intermediate values of a, soft-TTL is to be preferred, since it provides important gains over both TTL and fractional-TTL. Note, that the influence of a on TTL is studied in [16].

TABLE I
Optimal Utility Per File as a Function of Fairness Parameter α

α	W_1^{TTL}	W_3^{TTL}	W_1^{frac}	W_3^{frac}	W_1^{soft}	W_3^{soft}
0	0.1963	2.8335	0.4712	2.7913	0.3322	2.9262
0.5	0.4741	2.3872	0.5667	2.4602	0.6522	2.5391
2	0.8204	1.6057	0.8436	1.7578	0.8695	1.9709
∞	0.9215	1.3150	0.9999	0.9999	1.0000	1.1475

Finally, in Table I we demonstrate the impact of the fairness parameter α for the case of unequal arrivals. In particular, we set the arrival rates for the three files as $\lambda_1 = 1$, $\lambda_2 = 2$ and $\lambda_3 = 3$, and the other parameters as above. The table provides the optimal values of the utility functions W_1 and W_3 for the three policies. Beside the proportionality of the utilities in (8) to λ_i, we observe that, as the fairness parameter α grows larger, soft-TTL enables a large utility W_1 of the lower-rate file to be obtained, while also maintaining reasonably high utility W_3 of the higher-rate file, whereas fractional-TTL does so at the cost of largely reduced W_3.

References

[1] L. Wang, S. Bayhan, and J. Kangasharju, "Optimal chunking and partial caching in information-centric networks," *Comput. Commun.*, vol. 61, pp. 48–57, May 2015.
[2] A. Sengupta, R. Tandon, and O. Simeone, "Fog-aided wireless networks for content delivery: Fundamental latency tradeoffs," *IEEE Trans. Inf. Theory*, vol. 63, no. 10, pp. 6650–6678, Oct. 2017.
[3] M. Dehghan, L. Massoulie, D. Towsley, D. Menasche, and Y. C. Tay, "A utility optimization approach to network cache design," in *Proc. IEEE INFOCOM*, 2016, pp. 1–9.
[4] N. C. Fofack, P. Nain, G. Neglia, and D. Towsley, "Analysis of TTL-based cache networks," in *Proc. IEEE 6th Int. Conf. Perform. Eval. Methodol. Tools*, 2012, pp. 1–10.
[5] Q. Yang, M. M. Amiri, and D. Gündüz, "Audience retention rate aware coded video caching," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, 2017, pp. 1189–1194.
[6] E. Altman, K. Avrachenkov, and J. Goseling, "Distributed storage in the plane," in *Proc. IEEE Netw. Conf. (IFIP)*, 2014, pp. 1–9.
[7] M. Leconte *et al.*, "Placing dynamic content in caches with small population," in *Proc. IEEE INFOCOM*, 2016, pp. 1–9.
[8] A. Feldmann *et al.*, "Characteristics of TCP connection arrivals," in *Self-Similar Network Traffic and Performance Evaluation*. New York, NY, USA: Wiley, 2000, pp. 367–399.
[9] J. Jung, A. W. Berger, and H. Balakrishnan, "Modeling TTL-based Internet caches," in *Proc. IEEE INFOCOM*, 2003, pp. 417–426.
[10] M. Rausand and A. Høyland, *System Reliability Theory: Models, Statistical Methods, and Applications*. Hoboken, NJ, USA: Wiley, 2004.
[11] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Trans. Netw.*, vol. 8, no. 5, pp. 556–567, Oct. 2000.
[12] S. M. Ross, *Stochastic Processes*. Newburyport, MA, USA: Wiley, 1996.
[13] Gurobi Optimization, Inc. (2018). *Gurobi Optimizer Reference Manual*. [Online]. Available: http://www.gurobi.com
[14] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.
[15] K. M. Bretthauer and B. Shetty, "The nonlinear resource allocation problem," *Oper. Res.*, vol. 43, no. 4, pp. 670–683, 1995.
[16] A. Ferragut, I. Rodriguez, and F. Paganini, "Optimizing TTL caches under heavy-tailed demands," *SIGMETRICS Perform. Eval. Rev.*, vol. 44, no. 1, pp. 101–112, Jun. 2016.