Age of Information Aware Cache Updating with File-and Age-Dependent Update Durations

Haoyue Tang, Philippe Ciblat, Jintao Wang, Michèle Wigger, Roy Yates

Age of Information Aware Cache Updating with File- and Age-Dependent Update Durations

Haoyue Tang[1], Philippe Ciblat[3], Jintao Wang[1,2], Michèle Wigger[3], Roy Yates[4]
[1]Beijing National Research Center for Information Science and Technology (BNRist),
Dept. of Electronic Engineering, Tsinghua University, Beijing, China
[2]Research Institute of Tsinghua University in Shenzhen, Shenzhen, China
[3]Telecom Paris, Institut Polytechnique de Paris, Paris, France
[4]Rutgers University, New Brunswick, NJ, USA
{thy17@mails, wangjintao@}tsinghua.edu.cn, {ciblat, wigger}@telecom-paristech.fr, ryates@rutgers.edu

***Abstract*—We consider a system consisting of a library of time-varying files, a server that at all times observes the current version of all files, and a cache that at the beginning stores the current versions of all files but afterwards has to update these files from the server. Unlike previous works, the update duration is not constant but depends on the file and its *Age of Information (AoI)*, i.e., of the time elapsed since it was last updated. The goal of this work is to design an update policy that minimizes the average AoI of all files with respect to a given popularity distribution. Actually a relaxed problem, close to the original optimization problem, is solved and a practical update policy is derived. The update policy relies on the file popularity and on the functions that characterize the update durations of the files depending on their AoI. Numerical simulations show a significant improvement of this new update policy compared to the so-called square-root policy that is optimal under file-independent and constant update durations.**

I. Introduction

Caching, i.e., prestoring popular contents in cache memories close to end users, has become a popular tool to reduce congestion and latency in communication networks. For files that are both time-varying and time-sensitive, i.e., users wish to access recent versions of the files, the files have to be updated regularly. The size of an update thereby depends on the original size of the file and on the time elapsed since its last update, i.e., on the file's *Age of Information (AoI)*. In fact, if a file has been updated recently, then the data has not changed significantly, and its update is small. Instead, if a file has not been updated for a long time, then most of it needs to be replaced and the update is large. The main contribution of this work is to take account for this by letting the update durations depend on the file and, more importantly, on the file's current AoI.

Since the seminal paper [1], various AoI-related optimization problems have been studied recently for different systems configuration, see e.g. [2]–[7]. In this paper, we consider a single-server single-cache system where the remote server stores the current version of all files and the cache user updates the files in its memory by downloading fresh versions from this server. The goal is to minimize the average AoI of all the files in the cache when the average is taken with respect to a fixed popularity distribution. The current work assumes a non-stochastic setup similar to [8]–[10] where the remote server (which in practice may be distributed in the cloud) can always access the current version of all the files but the link between the remote server (or the cloud) and the cache is band-limited. Unlike [8], the update duration depends on the file and its age.

The contributions of the paper are as follows. We first formulate the optimization problem of minimizing the average AoI under AoI-dependent update durations. Then, we slightly relax and simplify the problem and solve this simplified version. Inspired by the solution, we propose a new practical cache update policy that respects all the original constraints. Through numerical simulations, we finally characterize the gain of this policy over the square-root policy obtained in [8].

The remainder of the paper is organized as follows. Section II states both the original and the simplified optimization problem. Section III solves the relaxed optimization problem through monotonic optimization and convex optimization theory. Section IV provides a practical model for the update duration. Section V presents a practical downloading policy inspired by the solution of the simplified optimization problem. Section VI presents numerical simulations and Section VII finally concludes the paper.

II. Problem formulation

A. System setup

The system consists of a remote server and a local server as depicted in Fig. 1. The remote server has a real-time access to N time-varying and time-sensitive files. The local server starts at time $t = 0$ with a fresh version of all files in its cache memory, and given the time-sensitivity of the files it wishes to keep the files as up-to-date as possible. It will therefore download and update fresh versions of the files from the remote server, where at any given time it can only update a single file. It is also assumed that a new file update can be started only once the previous updates are completed.

We consider the observation window $(0, T]$, for a fixed and given $T > 0$. Let K_n, for $n \in \{1, \dots, N\}$, denote the number

This work was supported by the ERC (Grant No. 715111), the National Key R&D Program of China (Grant No.2017YFE011230), Shenzhen basic Research Project (No.JCYJ20170816152246879) and the Tsinghua University Tutor Research Fund.

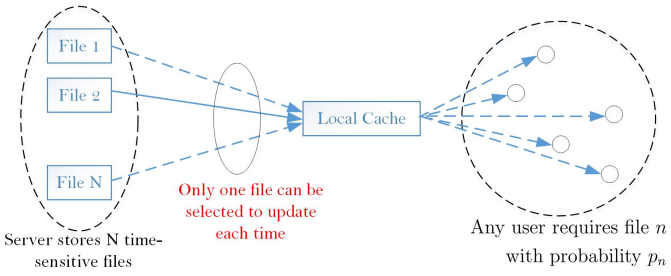

Fig. 1. The server-cache-users model.

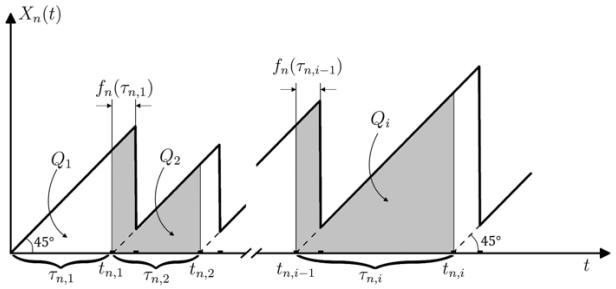

Fig. 2. A sample path of AoI evolution $X_n(t)$ for file n.

of updates of file n in this interval, and $0 < t_{n,1} < \cdots < t_{n,K_n}$ the *starting times* of these updates. Define then the inter-update intervals

$$\tau_{n,i} := \begin{cases} t_{n,1}, & i = 1; \\ t_{n,i} - t_{n,i-1}, & i = 2, \cdots, K_n; \\ T - t_{n,K_n}, & i = K_n + 1. \end{cases} \tag{1}$$

Given are also N functions $f_1(\cdot), \ldots, f_N(\cdot)$ which describe the time that it takes to update the different files, as will be made more clear in the following. Later in this paper we restrict to specific choices of these functions, but for now we only impose the following assumption:

Assumption 1 (Update function): Each function $f_n(\cdot)$ is assumed strictly positive, bounded, non-decreasing, concave, and differentiable over $\mathbb{R}_+$.

We now explain the update procedures and the associated age of information (AoI) process $\{X_n(t)\}_{t\geq 0}$ of the files. At time $t = 0$ the cache contains fresh versions of all the files, and hence the AoI of each file is $X_n(0) = 0$. The AoI of a given file n then grows as $X_n(t) = t$, until the cache has *finished* downloading a fresh version of this file. The cache starts the first update of file n at time $t_{n,1}$ and the update is finished at time $t_{n,1} + d_{n,1}$, where $d_{n,1}$ denotes the first update duration of file n. This update duration depends on the file's AoI at time $t_{n,1}$ and the update duration function f_n:

$$d_{n,1} := f_n(X_n(t_{n,1})) = f_n(t_{n,1}) = f_n(\tau_{n,1}), \tag{2}$$

where the equalities hold because the AoI at time $t_{n,1}$ is $X_n(t_{n,1}) = t_{n,1}$ and by Eq. (1).

At the *end* of this first update, at time $t = t_{n,1} + d_{n,1}$, the AoI of file n drops to the time elapsed since the fresh version was created, i.e., to $d_{n,1}$, and then grows again as $X_n(t) = t - t_{n,1}$ until the cache has *finished* the second update of the file. This second update starts at time $t_{n,2}$ and finishes at time $t_{n,2} + d_{n,2} = t_{n,2} + f_n(\tau_{n,2})$. When the second update ends, i.e., at time $t = t_{n,2} + d_{n,2}$, the AoI drops to $d_{n,2}$ and then grows as $X_n(t) = t - t_{n,2}$, and so forth. A sample path of the AoI is depicted in Fig 2. To summarize, the AoI of file $n \in \{1, \ldots, N\}$ is:

$$X_n(t) = \begin{cases} t\ , & t \in \big[0, t_{n,1} + d_{n,1}\big); \\ (t - t_{n,k})\ , & t \in \big[t_{n,k} + d_{n,k}, \\ & \qquad t_{n,k+1} + d_{n,k+1}\big), \end{cases} \tag{3}$$

where

$$d_{n,k} := f_n(\tau_{n,k}), \tag{4}$$

and the average AoI of a given file n is:

$$\overline{X}_n = \frac{1}{T} \int_0^T X_n(t) dt. \tag{5}$$

The main focus of this paper is on the expected AoI $\mathbb{X}$ over a file that is randomly chosen from the set $\{1, \ldots, N\}$ according to a given file popularity distribution $p_1, \ldots, p_N$ considered constant over a sufficiently large observation window,

$$\mathbb{X} := \sum_{n=1}^{N} p_n \overline{X}_n. \tag{6}$$

The update times $\{t_{n,1}, \ldots, t_{n,N}\}_{n=1}^N$ have to ensure that at each point in time only a single file is being updated. This is equivalent to requiring that the intervals $[t_{n,i}, t_{n,i} + d_{n,i})$ are disjoint for all $n \in \{1, \ldots, N\}$ and $i \in \{1, \ldots, K_n\}$.

B. Optimization problem

The goal is to minimize the expected AoI $\mathbb{X}$ in (6) over all feasible update times $\{t_{n,1}, \ldots, t_{n,K_n}\}_{n=1}^N$. We exploit the one-to-one correspondence in (1) and the functional relationship in (4) to express the optimization problem in terms of the inter-update intervals $\{\tau_{n,1}, \ldots, \tau_{n,K_n+1}\}_n$. Moreover, we simplify the cost function by splitting the integral in the computation of $\overline{X}_n$ into $K_n + 1$ subintervals:

$$\overline{X}_n = \frac{1}{T} \sum_{i=1}^{K_n+1} \int_{t=t_{n,i-1}}^{t_{n,i}} X_n(t) dt = \frac{1}{T} \sum_{i=1}^{K_n+1} Q_{n,i}, \tag{7}$$

where

$$Q_{n,i} := \int_{t_{n,i-1}}^{t=t_{n,i}} X_n(t) dt = \tau_{n,i-1} \cdot f_n(\tau_{n,i-1}) + \frac{1}{2} \tau_{n,i}^2. \tag{8}$$

Here the second equality holds because the integral corresponds to the sum of a parallelogram and a triangle as depicted by the gray area in Fig. 2.

We reach the following optimization problem.

Problem 1 (Original problem):

$$\min_{\{K_n, \tau_{n,i}\}} \sum_{n=1}^{N} p_n \left(\sum_{i=1}^{K_n+1} \frac{1}{2} \tau_{n,i}^2 + \sum_{i=1}^{K_n} \tau_{n,i} \cdot f_n(\tau_{n,i}) \right) \tag{9a}$$

where the minimum is over positive integers K_n and positive real numbers $\tau_{n,i}$ that satisfy the following two conditions:

$$\sum_{i=1}^{K_n+1} \tau_{n,i} = T, \quad \forall n \in \{1, \ldots, N\}, \tag{9b}$$

and for all i, j, n, n' satisfying $(n, i) \neq (n', j)$:

$$\left(\sum_{k=1}^{i} \tau_{n,k} - \sum_{k=1}^{j} \tau_{n',k} \right) \notin \Big(-f_n(\tau_{n,i}),\ f_{n'}(\tau_{n',j}) \Big]. \tag{9c}$$

Condition (9c) holds because the update intervals $\big[\sum_{k=1}^{i} \tau_{n,k}, \sum_{k=1}^{i} \tau_{n,k} + f_n(\tau_{n,i})\big)$ and $\big[\sum_{k=1}^{j} \tau_{n',k}, \sum_{k=1}^{j} \tau_{n',k} + f_{n'}(\tau_{n',j})\big)$ have to be disjoint.

III. A Relaxed Suboptimal Solution

The presented optimization problem seems hard, even in the asymptotic regime $T \to \infty$, on which we will focus shortly. We therefore relax constraint (9c) into the next constraint

$$\sum_{n=1}^{N} \sum_{i=1}^{K_n} f_n(\tau_{n,i}) \leq T, \tag{10}$$

i.e., several files can be updated simultaneously as long as the global update duration remains smaller than the observation window. We also choose (possibly suboptimally) uniform inter-update intervals

$$\tau_{n,i} = \overline{\tau}_n := \frac{T}{K_n + 1}. \tag{11}$$

The optimization problem then consists in finding the optimal choices of $\{\overline{\tau}_n\}$, and is stated in Section III-A. To motivate the choice in (11), notice that it is optimal when the update durations are constant and identical across files [8]. Moreover, in Section V we present a practical update policy that updates only a single file at each time, and has a performance close to the solution to our new optimization problem.

A. A Relaxed Suboptimal Problem

In what follows, consider the asymptotic regime $T \to \infty$. By (9a), the expected AoI $\mathbb{X}$ grows without bound unless for all files n the number of updates $K_n \to \infty$ as $T \to \infty$. We can therefore assume in the following

$$\lim_{T \to \infty} \frac{K_n}{K_n + 1} = 1. \tag{12}$$

Plugging (11) and (12) into (9a) and (10) results in the following new optimization problem, which approximates the original problem in the asymptotic regime where $T \to \infty$.

Problem 2 (Relaxed suboptimal problem – version 1):

$$\min_{\{\overline{\tau}_n\}_n} \sum_{n=1}^{N} p_n \left(\frac{1}{2} \overline{\tau}_n + f_n(\overline{\tau}_n) \right) \tag{13a}$$

s.t. $\overline{\tau}_n \geq 0,\ \forall n$, and

$$\sum_{n=1}^{N} \frac{f_n(\overline{\tau}_n)}{\overline{\tau}_n} \leq 1. \tag{13b}$$

We reformulate the optimization in terms of the *file utilization ratios*, i.e., the fraction of time that each file is being updated,

$$\lambda_n := \frac{f_n(\overline{\tau}_n)}{\overline{\tau}_n}. \tag{14}$$

For finite T, this fraction is $(1/T) \sum_{i \in \{1, \cdots, K_n\}} f_n(\tau_{n,i}) = K_n f_n(\overline{\tau}_n) / ((K_n + 1)\overline{\tau}_n)$, which by (12) tends to $\frac{f_n(\overline{\tau}_n)}{\overline{\tau}_n}$ as $T \to \infty$. Due to Eq. (13b), we get $\lambda_n \leq 1$ for the feasible points of Problem 2, i.e., the duration spent for updating the n-th file $f_n(\overline{\tau}_n)$ is smaller than the inter-update duration $\overline{\tau}_n$.

We have the following useful lemma on the function

$$g_n(t) := \frac{f_n(t)}{t}. \tag{15}$$

Lemma 1: Under Assumption 1, the function $t \mapsto g_n(t)$ for $t \in \mathbb{R}_+$ is strictly decreasing and its image is $(0, \infty)$. Consequently, g_n has an inverse function denoted by $g_n^{(-1)}$, which is also strictly decreasing.

Proof: The derivative can be upper bounded as:

$$g_n'(t) = \frac{t f_n'(t) - f_n(t)}{t^2} \overset{(a)}{\leq} \frac{-f_n(0)}{t^2} \overset{(b)}{<} 0, \tag{16}$$

where the inequality (a) is due to the concavity of the function f_n and the inequality (b) is due to its positivity. So g_n is strictly decreasing. Since $\lim_{t \to 0} f_n(t) = f_n(0) > 0$, we have $\lim_{t \to 0} \frac{f_n(t)}{t} = \infty$. As the function f_n is upper-bounded, we also have $\lim_{t \to \infty} \frac{f_n(t)}{t} = 0$. Since f_n is differentiable (and so continuous) and strictly decreasing, its image is $(0, \infty)$. The rest of the proof is straightforward. ∎

Therefore, $\bar{\tau}_n = g_n^{(-1)}(\lambda_n)$, and Problem 2 is easily rewritten as an optimization problem over $\{\lambda_n\}_{n=1}^N$.

Problem 3 (Relaxed suboptimal problem – version 2): Let

$$h_n(\lambda) := g_n^{(-1)}(\lambda) \left(\frac{1}{2} + \lambda \right). \tag{17}$$

Problem 2 is equivalent to:

$$\min_{\{\lambda_n\}_n} \sum_{n=1}^{N} p_n \cdot h_n(\lambda_n) \tag{18a}$$

s.t. $\lambda_n \geq 0,\ \forall n$, and

$$\sum_{n=1}^{N} \lambda_n \leq 1. \tag{18b}$$

We finish this subsection by showing that the optimal solution must lie on the boundary of the feasible set. In subsequent subsections we discuss numerical optimization methods that can be used to solve our problem. We also present the KKT conditions, which can be used to simplify the search of the optimal solution when the function h_n is convex.

We have the following auxiliary lemma, which will be useful throughout the paper.

Lemma 2: The function $h_n(\cdot)$ is strictly decreasing over $\mathbb{R}_+$.

Proof: According to Proposition 1, $g_n^{(-1)}(\lambda)$ is strictly decreasing. In addition, thanks to Eq. (15), we get $f_n(g_n^{(-1)}(\lambda)) = g_n^{(-1)}(\lambda) \cdot g_n(g_n^{(-1)}(\lambda)) = g_n^{(-1)}(\lambda)\lambda$. Since

f_n is non decreasing and $g_n^{(-1)}$ is strictly decreasing, the composition $f_n \circ g_n^{(-1)}$ is a non increasing function. ∎

Proposition 1: Let $\{\lambda_n^\star\}_n$ be the optimal solution of Problem 3. It satisfies:

$$\sum_{n=1}^{N} \lambda_n^\star = 1, \tag{19}$$

i.e., it lies on the boundary of the feasible set.

Proof: By contradiction, let us assume $\sum_{n=1}^N \lambda_n^\star < 1$. Then for an arbitrary n_0, we replace $\lambda_{n_0}^\star$ with $\lambda_{n_0}^\star + \delta_{n_0}$ to force equality in Eq. (19). As h_n is strictly decreasing, we get $h_n(\lambda_{n_0}^\star + \delta_{n_0}) < h_n(\lambda_{n_0}^\star)$. And the point $\lambda_{n_0}^\dagger = \lambda_{n_0}^\star + \delta_{n_0}$ and $\lambda_n^\dagger = \lambda_n^\star$ for $n \neq n_0$ is better than the optimal one, which concludes the proof. ∎

B. Monotonic optimization solution

Problem 3 can be cast into the monotonic optimization framework [11]–[13] because the constraints are linear (and thus convex) and the cost function is strictly decreasing by Lemma 2. The optimal solution can thus be found using the so-called Branch-Reduce-Bound (BRB) [13]. Notice that the function $h_n(\lambda)$ grows without bound when $\lambda \to 0$ (In this limit the file is not updated and its age diverges, see Eq. (26).). The BRB algorithm therefore has to remove a tiny neighborhood around the origin from the initially selected box.

C. KKT based algorithm

The function h_n is determined by the update function f_n and is generally not convex. This makes that in general the Karush-Kuhn-Tucker (KKT) conditions are only necessary but not sufficient for an optimal solution. However, for many practically relevant choices of the update function f_n (see Section IV for more details), the function h_n is convex. We therefore derive the KKT conditions in this subsection.

Let us define the Lagrangian

$$\mathcal{L}(\lambda_1, \ldots, \lambda_N, \nu) = \sum_{n=1}^{N} h_n(\lambda_n) + \nu \left(\sum_{n=1}^{N} \lambda_k - 1 \right)$$

with $\nu \geq 0$ the Lagrange multiplier. The KKT conditions then state that the primal-dual optimal solutions $(\lambda_1^\star, \cdots, \lambda_N^\star, \nu^\star)$ must satisfy

$$p_n h_n'(\lambda_n^\star) + \nu^\star = 0, \quad \forall\ n \tag{20}$$

$$\nu^\star \left(\sum_{n=1}^{N} \lambda_n^\star - 1 \right) = 0, \tag{21}$$

where h_n' denotes the first-order derivative of h_n:

$$h_n'(\lambda) := {g_n^{(-1)}}'(\lambda) \left(\frac{1}{2} + \lambda \right) + g_n^{-1}(\lambda). \tag{22}$$

Notice that h_n' is invertible, and the image of this inverse $h_n'^{(-1)}$ is the set of all non-positive real numbers $(-\infty, 0]$. This latter property holds because h_n is differentially continuous and goes from $+\infty$ to 0. We can thus rewrite (20) as

$$\lambda_n^\star = h_n'^{(-1)} \left(-\frac{\nu^\star}{p_n} \right), \quad \forall n. \tag{23}$$

Condition (21) is subsumed by the stronger condition $\sum_{n=1}^N \lambda_n^\star = 1$, which was proved in Proposition 1. The optimal primal variables $\lambda_1^\star, \ldots, \lambda_N^\star$ are thus given by (23) for some "waterlevel" $\nu^\star \geq 0$ that needs to be chosen so that $\sum_{n=1}^N \lambda_n^\star = 1$. Certain functions f_n permit to find a closed-form expression for $h_n'^{(-1)}$. For other functions one needs to search over the entries of a Look Up Table to find the desired values of $h_n'^{(-1)}$.

Sometimes it is more convenient to perform a change of variables and express the KKT conditions in terms of the optimal inter-update intervals $\overline{\tau}_n^\star = g_n^{(-1)}(\lambda_n^\star)$. Since, ${g_n^{(-1)}}'(\lambda_n^\star) = \frac{1}{g_n'(\tau_n^\star)} = \frac{(\overline{\tau}_n^\star)^2}{f_n'(\overline{\tau}_n^\star)\overline{\tau}_n^\star - f_n(\overline{\tau}_n^\star)}$, Eq. (23) is equivalent to

$$\overline{\tau}_n^\star + \frac{(\overline{\tau}_n^\star)^2}{f_n'(\overline{\tau}_n^\star)\overline{\tau}_n^\star - f_n(\overline{\tau}_n^\star)} \left(\frac{1}{2} + \frac{f_n(\overline{\tau}_n^\star)}{\overline{\tau}_n^\star} \right) = -\frac{\nu^\star}{p_n}, \forall n, \tag{24}$$

where f_n' denotes the derivative of f_n. This equation can be easier to solve because it does not include the inverse $g_n^{(-1)}$. For instance, for $f_n(t) = B_n - (B_n - \varepsilon_n)/(1+t)$ (with $B_n > \varepsilon_n > 0$ well tuned in order to ensure the convexity of h_n), solving Eq (24) is equivalent to finding the positive real-valued root of a fourth-order polynomial and thus a closed-form solution exists.

IV. A PRACTICAL EXAMPLE FOR f_n

In this section, we motivate a specific choice of f_n, which will extensively be used in the simulation part. The idea is that in each unit of time, a certain portion of each file becomes obsolete, and that the cache and the server know the obsolete parts. These bits can thus be modeled as erasures, and we model the evolution of file n as passing each bit through a Binary Erasure Channel (BEC) with parameter Δ_n. Assume that the file n initially consists of B_n bits. Then, after t time units without update, any given bit of the file n undergoes t sequential applications of a BEC with parameter Δ_n. This transition can be modeled as a BEC with parameter $1-(1-\Delta_n)^t$, and the average number of erased positions in the file after t time units is $B_n(1-(1-\Delta_n)^t)$.

However, when $\lim_{t\to 0} f_n(t) = 0$, then degenerate solutions like $\bar{\tau}_n^\star = 0$ could be optimal, which is not feasible in practice. We therefore add an offset ε_n to each update function. Combined with the arguments in the previous paragraph, we obtain $f_n(t) = B_n - (B_n - \varepsilon_n)(1-\Delta_n)^t$ or expressed in an exponential form:

$$f_n(t) = B_n - (B_n - \varepsilon_n) e^{-\beta_n t} \tag{25}$$

with $\beta_n = -\log(1-\Delta_n)$. Notice that $\beta_n > 0$ and that such a function f_n always satisfies Assumption 1.

For the choice in Eq. (25),

$$g_n^{(-1)}(\lambda) = \frac{B_n}{\lambda} + \frac{1}{\beta_n} W\left(-\frac{\beta_n(B_n - \varepsilon_n)}{\lambda} e^{-\frac{\beta_n B_n}{\lambda}} \right), \tag{26}$$

where $W(\cdot)$ denotes the Lambert function. Notice that the associated function $h_n = g_n^{(-1)}(\lambda)(\frac{1}{2}+\lambda)$ is convex for certain values of ϵ_n, B_n, β_n (for instance, for the set of parameters selected in Section VI). In this case the solution can be found

based on the KKT conditions. For other values of ϵ_n, B_n, β_n (for instance, for $B_n = 1$, $\varepsilon_n = 0.02$, and $\beta_n = 10$) the function h_n is non-convex and we suggest to use the BRB algorithm to find the optimal solution for the relaxed problem.

V. A Practical Scheduling Algorithm

The question now is: how to apply the result of the previous sections to obtain a practical scheduling algorithm that satisfies all the original constraints? In particular, only a single update should be scheduled at any given point in time, and a new update can only start once the previous update has terminated.

To describe the practical update algorithm, let $\{\lambda_n^\star\}$ be an optimal solution to Problem 3, which is either obtained with the BRB algorithm or with the KKT-based algorithm (if h_n is convex). Then set $\overline{\tau}_n^\star := g_n^{(-1)}(\lambda_n^\star)$. If the algorithm has to schedule a new update at a given time t (because the previous file update just finished), it will choose the file that is currently most urgent, i.e., whose AoI is closest to its maximum target AoI $\overline{\tau}_n^\star$. More precisely, the algorithm schedules any of the files $n_0(t) \in \{1, \ldots, N\}$ that satisfies

$$n_0(t) = \arg \min_{n \in \{1, \cdots, N\}} \overline{\tau}_n^\star - X_n(t).$$

VI. Simulations

In this Section, we numerically compare our idealized and practical scheduling policies with the so-called square-root (sqrt) strategy developed in [8], which is optimal when the update duration equals the same constant value for all files whose the utilization ratio is given by

$$\lambda_n^\star = \lambda_n^{\text{sqrt}} = \frac{\sqrt{p_n}}{\sum_{i=1}^N \sqrt{p_i}}. \tag{27}$$

We present numerical simulations for two choices of the update functions: *i)* $f_n(t) = B_n$, and *ii)* $f_n(t)$ given in Eq. (25). In all the following figures, blue curves indicate the performance under constant identical update durations and orange curves the performance under one of the two choices of functions $\{f_n\}$. Solid curves indicate the solutions of the relaxed problems (either Problem 3 for the proposed scheduling policy or the optimization problem described in [8]) and dashed curves correspond to the proposed practical scheduling algorithms (see Section V) that avoid collisions.

A. File-dependent but age-independent update durations

Assume $f_n(t) = B_n$. In this case, $g_n(t) = B_n/t$ and $g_n^{(-1)}(\lambda) = B_n/\lambda$ leading to:

$$h_n(\lambda) = \frac{B_n}{2\lambda} + B_n.$$

This function is convex, and we just need to solve the KKT conditions. According to Eq. (23), we obtain

$$\lambda_n^\star = \frac{\sqrt{p_n B_n}}{\sum_{i=1}^N \sqrt{p_i B_i}}. \tag{28}$$

The work in [8] considered update ratios as optimization parameters and not utilization ratios. For constant update durations the two notions coincide. Notice that the policy given in Eq. (28) is a slight modification of the one given in Eq. (27) by weighting the popularity of a file with its update duration. According to Eq. (28), for two files having the same update duration, the most popular one will be updated more often. For two files with the same popularity, the file with longer update duration will get a larger utilization ratio. However, as the update ratio (proportion of updates done within the observation window) is equal to λ_n/B_n, the files with longer update duration will be updated less frequently.

We split the files into two categories: for $n \in \{1, 2, \cdots, N/2\}$, $B_n = 1$ with popularity $p_n \propto 1/n^\alpha$; for $n \in \{N/2+1, 2, \cdots, N\}$, $B_n = 5$ and $p_n = p_{n-N/2} \propto 1/(n-N/2)^\alpha$. Each category thus obeys a Zipf-like distribution with parameter α. We fix $\alpha = 1.8$.

In Fig. 3, we plot the average AoI versus the number of files N. We observe that the proposed strategy outperforms the square root law based strategy. For instance, the gain is 10% at $N = 50$. Moreover the loss in performance of the practical algorithm (which prevents from collision, i.e., only one file is scheduled) is small compared to the relaxed solution (which does not prevent to schedule multiple files).

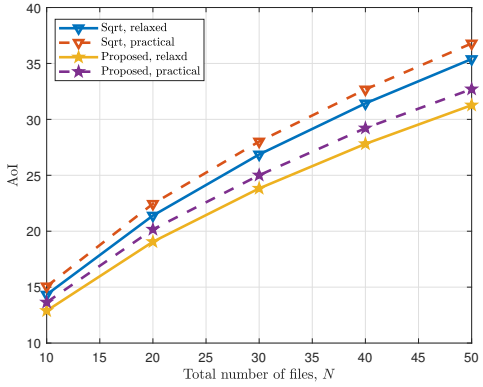

Fig. 3. Average AoI versus N (non-identical but constant update durations).

B. Age-dependent update durations

Throughout this section, assume f_n as in Eq. (25) and fix $\varepsilon_n = 0.02, B_n = 1$, for all files n. We will consider different values for the parameters β_n. In particular, we consider setups where all β_ns are the same, and thus the update durations depend only on a file's age but not on the identity (index) of the file, and setups with different β_ns. Throughout this section, we assume that the popularity of the files follows a Zipf-distribution with parameter $\alpha = 1.8$, and we apply the BRB algorithm as described in Section III-B to find the optimal solution.

We consider $\beta_n = 0.015$, $\forall n$, so all the files have the same update function. In Fig. 4, we plot the average AoI versus N. The proposed strategy achieves a smaller AoI compared to the square root law. For $N = 5$, the gain is around 50% for the practical algorithm.

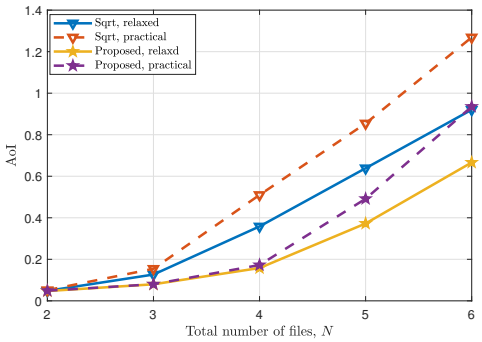

Fig. 4. Average AoI versus N (non-constant update durations).

In Fig. 5, we plot the individual utilization ratios versus the file indices (sorted by popularity's order) when $N = 5$. We observe that the optimal utilization ratio significantly differs from the square root law. Actually, the most popular files are updated more frequently and their update durations are shorter. This finally leads to a smaller utilization ratio for the most popular files than those obtained with the square-root law..

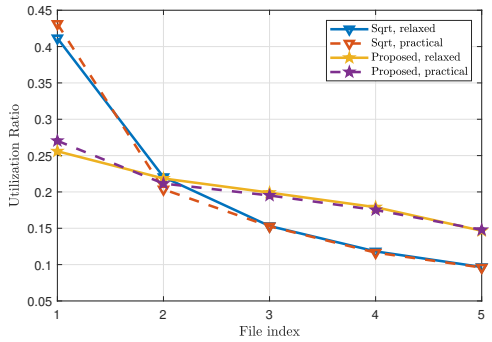

Fig. 5. Utilization ratio versus file index (non-constant update durations).

Now, all files have same popularity ($p_n = 1/N$, $\forall n$) but different update duration function with $\beta_n = 0.1 \times n$, $\forall n$. In Fig. 6, we plot the AoI and the utilization ratio versus the file index for $N = 5$. For large values of n, the parameter β_n is also large and the update function f_n in Eq. (25) is almost constant. For these files, the utilization ratios of the proposed algorithms are close to the ones under the square root law (which is optimal under constant update durations). Instead for small n the parameter β_n is small and the update function f_n is strictly increasing for small AoIs. For these files the utilization ratios of our algorithms are significantly smaller than under the square-root law. A closer inspection of our simulations reveals that these files are updated frequently, but each update is short. For moderate n the utilization ratio is large because these are updated frequently and each update is not very short.

VII. Conclusion

This paper proposes a practical algorithm for scheduling updates from a remote server to a local cache when the update duration depends on the file's AoI. The proposed algorithm is shown to have small performance loss compared to the optimal scheduling policy of a relaxed problem. In all these results, a given file popularity is taken into account.

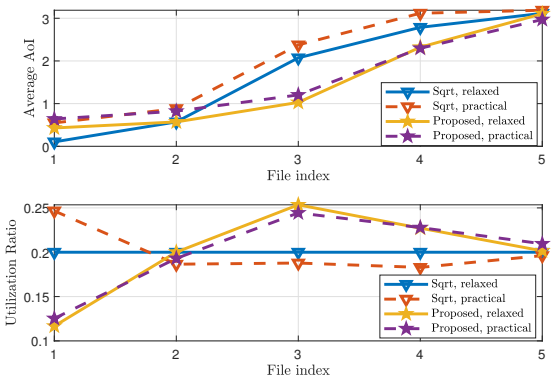

Fig. 6. Average AoI (top) and Utilization ratio (bottom) versus file index (non-identical and non-constant update durations).

References

[1] S. Kaul, R. Yates, and M. Gruteser, "Real-time status: How often should one update?" in *2012 Proceedings IEEE INFOCOM*, March 2012, pp. 2731–2735.

[2] R. D. Yates and S. K. Kaul, "The age of information: Real-time status updating by multiple sources," *IEEE Transactions on Information Theory*, vol. 65, no. 3, pp. 1807–1827, March 2019.

[3] C. Kam, S. Kompella, G. D. Nguyen, J. E. Wieselthier, and A. Ephremides, "Information freshness and popularity in mobile caching," in *2017 IEEE International Symposium on Information Theory (ISIT)*, June 2017, pp. 136–140.

[4] B. Zhou and W. Saad, "Optimal sampling and updating for minimizing age of information in the internet of things," in *2018 IEEE Global Communications Conference (GLOBECOM)*, Dec 2018, pp. 1–6.

[5] C. R. Talak, I. Kadota, S. Karaman, and E. Modiano, "Scheduling policies for age minimization in wireless networks with unknown channel state," in *2018 IEEE International Symposium on Information Theory (ISIT)*, June 2018, pp. 2564–2568.

[6] I. Kadota, A. Sinha, E. Uysal-Biyikoglu, R. Singh, and E. Modiano, "Scheduling policies for minimizing age of information in broadcast wireless networks," *Arxiv*, vol. abs/1801.01803, 2018.

[7] B. Wang, S. Feng, and J. Yang, "When to preempt? age of information minimization under link capacity constraint," *Journal of Communications and Networks*, vol. 21, pp. 220–232, June 2019.

[8] R. D. Yates, P. Ciblat, A. Yener, and M. Wigger, "Age-optimal constrained cache updating," in *2017 IEEE International Symposium on Information Theory (ISIT)*, June 2017, pp. 141–145.

[9] M. Bastopcu and S. Ulukus, "Age of Information for Updates with Distortion," in *IEEE Information Theory Workshop (ITW)*, Aug. 2019.

[10] G. Ahani, D. Yuan, and S. Sun, "Optimal scheduling of age-centric cacinhg: tractability and computation," *submitted to IEEE Trans. on Mobile Computing (arxiv/2001.00259)*, 2020.

[11] E. Jorswieck and E. Larsson, "Monotonic optimization framework for the two-user MISO interference channel," *IEEE Transactions on Communications*, vol. 58, no. 7, pp. 2159–2169, July 2010.

[12] E. Björnson, G. Zheng, M. Bengtsson, and B. Ottersten, "Robust monotonic optimization framework for multicell MISO systems," *IEEE Transactions on Signal Processing*, vol. 60, no. 5, pp. 1–16, May 2012.

[13] E. Björnson and E. Jorswieck, *Optimal Resource Allocation in Coordinated Multi-Cell Systems*. Now Publishers, 2013, vol. 9, no. 2-3, pp. 113–381.