Freshness Based Cache Updating in Parallel Relay Networks

Priyanka Kaswan Melih Bastopcu Sennur Ulukus
Department of Electrical and Computer Engineering
University of Maryland, College Park, MD 20742
pkaswan@umd.edu *bastopcu@umd.edu* *ulukus@umd.edu*

2021 IEEE International Symposium on Information Theory (ISIT) | 978-1-5386-8209-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/ISIT45174.2021.9518003

***Abstract*—We consider a system consisting of a server, which receives updates for N files according to independent Poisson processes. The goal of the server is to deliver the latest version of the files to the user through a parallel network of K caches. We consider an update received by the user successful, if the user receives the same file version that is currently prevailing at the server. We derive an analytical expression for information freshness at the user. We observe that freshness for a file increases with increase in consolidation of rates across caches. To solve the multi-cache problem, we first solve the auxiliary problem of a single-cache system. We then rework this auxiliary solution to our parallel-cache network by consolidating rates to single routes as much as possible. This yields an approximate (sub-optimal) solution for the original problem. We provide an upper bound on the gap between the sub-optimal solution and the optimal solution. Numerical results show that the sub-optimal policy closely approximates the optimal policy.**

I. Introduction

In the information age, users want instant access to up-to-date data. Caching is a popular method of pre-storing data at nodes in a network closer to the users for faster delivery of latest data. In recent years, various papers have explored freshness-optimal policies in different settings. Most works have relied on the age of information (AoI) metric to measure obsoleteness of data. AoI has been considered in a wide range of contexts, such as queueing networks, energy harvesting systems, web crawling, scheduling problems, remote estimation, UAV systems and so on [1]–[49].

The works that are most closely related to our work here are [40]–[49]. In [40], a single-server single-cache refresh system is considered, where it is shown that an asymptotically optimal policy updates a cached file in proportion to the square root of its popularity. The work in [40] assumes constant file update durations, which is extended in [41] by considering file update durations to be dependent on the size and the age of the files. While [40], [41] use the AoI metric, reference [42] uses a binary freshness metric in a caching system, and determines the optimum update rates at the user and the cache. [42] also extends the approach to a cascade sequence of cache nodes, and [43] generalizes it to the case of nodes with limited cache capacity. Here, we further generalize [42] to a more complex network which is composed of parallel caches.

Other related work that use caching and relaying techniques for freshness include: [44] where a tradeoff between content freshness and service latency from the aspect of mobile edge caching is studied; [45] which considers caching policies in opportunistic networks; [46] where a cache-enabled aggregator decides whether to receive a fresh update from an energy harvesting sensor or serve the request with a cached update; [47] where an optimal policy is derived when the current rate of requests for a file is dependent on both history of requests and the freshness of the file; [48] which considers a two-hop status update system where an optimal scheduling policy is identified by a constrained Markov decision process approach; and [49] where a two-hop system with energy harvesting at source and relay nodes is considered.

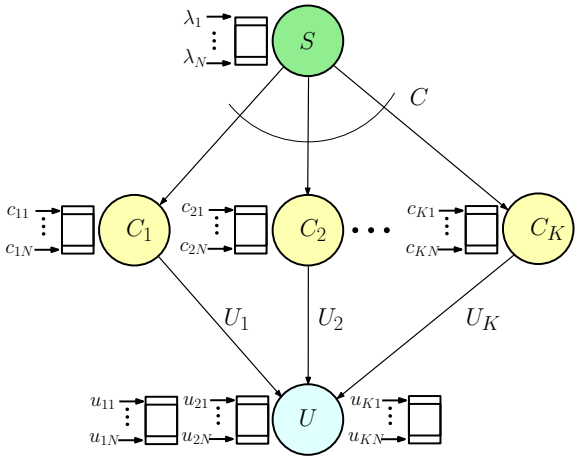


Fig. 1. System model for a parallel multi-cache system.

In this paper, we consider a parallel network with multiple cache routes between a source and a user (Fig. 1). We first derive a closed-form expression for freshness at the user. We observe from the freshness formula of the two-cache system that lop-sided distribution of rates across the routes supports higher freshness. Further, for the two-route two-file case, restricting at least one of the files to a single route maximizes the overall freshness of the system. Moreover, in a K-cache system, restricting a file to fewer routes improves the freshness. Motivated by these properties, we solve an auxiliary problem of a single-cache system and adapt its solution to our parallel-cache network to obtain an approximate (sub-optimal) solution for the original problem. We provide an upper bound on the gap between the sub-optimal policy and the optimal policy. The gap is finite and is independent of the number of files. Numerical results show that the proposed sub-optimal policy closely approximates the optimal policy.

II. System Model and Problem Formulation

We consider a system with a source, K parallel relays and a user, as shown in Fig. 1. The source has the most up-to-date versions of a library of N files. File i is updated at the source with exponential inter-update times with rate λ_i. The source updates file i at cache k with exponential inter-update times with rate c_{ki}. Cache k updates file i at the user with exponential inter-update times with rate u_{ki}. There is no delay or information loss in any source-cache links or cache-user links. However, the source is subject to a total update rate constraint $\sum_{k=1}^{K}\sum_{i=1}^{N} c_{ki} \leq C$, and cache k is subject to a total update rate constraint $\sum_{i=1}^{N} u_{ki} \leq U_k$, for $k = 1, \ldots, K$.

When a file is updated at the source, the stored versions of the same file at the caches and at the user become outdated. Thus, we consider an update received by the user successful if the user receives a file version that is currently prevailing at the source. This will happen when the source updates the cache and the cache in turn updates the user before the file at the source is updated with a newer version. In the following subsections, we first derive a freshness expression for file i in a single-cache system, and then in a multi-cache system. For simplicity, we drop subscript i from λ_i, c_{ki} and u_{ki} since the derivation is valid for all files (for all i).

A. Freshness of File i in the Single-Cache Model

In this subsection, we find the freshness expression for file i for a single-cache system. First, we characterize the freshness at the cache. In Fig. 2(a), the freshness evolution at the cache is shown between two file updates at the source. We define the freshness function for file i at the cache as follows

$$f_c(i,t) = \begin{cases} 1, & \text{if file } i \text{ at the cache is fresh at time } t, \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

Let $T_s(i,j)$ denote the jth update cycle at the source, i.e., time interval between the jth and $(j+1)$th update for file i. Once the source gets updated, the cache is updated after duration $W_c(i,j)$ and it remains updated for $T_c(i,j) = T_s(i,j) - W_c(i,j)$ duration. For simplicity, we drop index i for variables $T_s(i,j)$, $T_c(i,j)$, and $W_c(i,j)$, as the results in this subsection pertain to file i. We denote $F_c(i)$ as the long term average freshness of file i at the cache which is given by

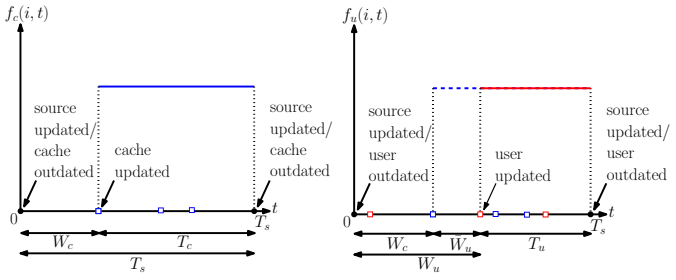
$$F_c(i) = \lim_{T\to\infty} \frac{1}{T}\int_0^T f_c(i,t)dt. \quad (2)$$

Let M be the number of update cycles in time duration T. Provided that the system is ergodic, similar to [42], $F_c(i)$ is

$$F_c(i) = \lim_{T\to\infty} \frac{M}{T}\left(\frac{1}{M}\sum_{j=1}^{M} T_c(j)\right) = \frac{\mathbb{E}[T_c]}{\mathbb{E}[T_s]}. \quad (3)$$

Here, as $T_c(j)$ are independent and identically distributed (i.i.d.) over j, we drop the index j and denote $T_c(j)$ with the typical random variable T_c. Similarly, T_s and W_c denote the typical random variables for $T_s(j)$ and $W_c(j)$, respectively.

Since T_s is an exponential random variable with rate λ, we have $\mathbb{E}[T_s] = \frac{1}{\lambda}$. We find $\mathbb{E}[W_c]$ by using nested expectations,



(a) (b)

Fig. 2. Freshness function as a function of time at the cache and the user.

i.e., $\mathbb{E}[W_c] = \mathbb{E}[\mathbb{E}[W_c|T_s]]$. For a given update cycle duration $T_s = t$ at the source, W_c, which is exponentially distributed with rate c, either takes a value in between 0 and T_s, or takes value T_s, i.e., $W_c = T_s$, and in this case, the cache is not updated in that cycle. Thus, we have

$$\mathbb{E}[W_c|T_s = t] = \int_0^t xce^{-cx}dx + \int_t^\infty tce^{-cx}dx = \frac{1-e^{-ct}}{c}. \quad (4)$$

Then, we obtain $\mathbb{E}[W_c]$ as

$$\mathbb{E}[W_c] = \int_0^\infty \frac{1-e^{-ct}}{c}\lambda e^{-\lambda t}dt = \frac{1}{\lambda+c}. \quad (5)$$

By using (5), we obtain $\mathbb{E}[T_c] = \mathbb{E}[T_s] - \mathbb{E}[W_c]$ as

$$\mathbb{E}[T_c] = \frac{c}{\lambda(\lambda+c)}. \quad (6)$$

Finally, by substituting (6) into (3), we obtain $F_c(i)$ as

$$F_c(i) = \frac{c}{\lambda+c}. \quad (7)$$

Next, we characterize the freshness at the user. Freshness evolution at user in an update cycle is shown in Fig. 2(b). We define the freshness function for file i at the user as follows

$$f_u(i,t) = \begin{cases} 1, & \text{if file } i \text{ at the user is fresh at time } t, \\ 0, & \text{otherwise.} \end{cases} \quad (8)$$

Once file i is updated at the cache after $W_c(j)$, and the same file is updated at the user after $\bar{W}_u(j)$, file i at the user remains fresh for a time period of $T_u(j)$. Thus, the total waiting time for the user to get the freshest version of file i in the jth cycle is $W_u = W_c + \bar{W}_u$. We denote $F_u(i)$ as the long term average freshness of the file i at the user which is given by

$$F_u(i) = \frac{\mathbb{E}[T_u]}{\mathbb{E}[T_s]}, \quad (9)$$

where T_u denotes the typical random variable for $T_u(j)$.

First, we find $\mathbb{E}[W_u]$ by using nested expectations. When the jth update arrives at the source, due to memoryless property of the exponential distribution, W_c and $\bar{W}_u$ are exponentially distributed with rates c and u, respectively. Hence, the distribution of W_u denoted by $f_{W_u}(x)$ is equal to the convolution of the distributions of W_c and $\bar{W}_u$ which is given by

$$f_{W_u}(x) = \frac{cu}{c-u}\left(e^{-ux} - e^{-cx}\right), \quad 0 \leq x < \infty. \quad (10)$$

For a given update cycle duration $T_s = t$ at the source, the

total waiting time W_u with pdf in (10) either takes a value in between 0 and T_s, or $W_u = T_s$. When $W_u = T_s$, we note that the file at the user is not updated in that cycle. Thus,

$$\mathbb{E}[W_u|T_s = t] = \frac{cu}{c-u}\left(\frac{1-e^{-ut}}{u^2} - \frac{1-e^{-ct}}{c^2}\right). \quad (11)$$

By using $\mathbb{E}[W_u] = \mathbb{E}[\mathbb{E}[W_u|T_s]]$, we obtain $\mathbb{E}[W_u]$ as

$$\mathbb{E}[W_u] = \frac{\lambda + c + u}{(\lambda + u)(\lambda + c)}. \quad (12)$$

Then, we obtain $\mathbb{E}[T_u] = \mathbb{E}[T_s] - \mathbb{E}[W_u]$ as

$$\mathbb{E}[T_u] = \frac{uc}{\lambda(\lambda + u)(\lambda + c)}. \quad (13)$$

Finally, by substituting (13) into (9), we obtain $F_u(i)$ as

$$F_u(i) = \frac{\mathbb{E}[T_u]}{\mathbb{E}[T_s]} = \frac{u}{\lambda + u}\frac{c}{\lambda + c}, \quad (14)$$

which is equal to the freshness expression in [42]. Above, we have provided an alternative method (to [42]) to derive freshness, which will be useful in the multi-cache system next.

B. Freshness of File i in the Multi-Cache Model

In this subsection, we find the freshness expression of file i for a multi-cache system. For simplicity, we drop file index i from all variables. Each cache sends its updates to the user independent of other caches. After the file at the source is updated for the jth time, the file at the user becomes fresh again by the first successful update by any one of the caches. The file at the cache k is updated after W_{c_k} duration. The cache k updates the same file at the user after $\bar{W}_{u_k}$ duration. We denote the random variable $X_k = W_{c_k} + \bar{W}_{u_k}$ as the total waiting time for cache k to send a successful update to the user. As W_{c_k} and $\bar{W}_{u_k}$ are exponentially distributed with rates c_k and u_k, respectively, similar to (10), we have $f_{X_k}(x) = \frac{c_k u_k}{c_k - u_k}(e^{-u_k x} - e^{-c_k x})$ for $x \geq 0$. For a given update cycle $T_s = t$, the user is updated after W_u given by

$$W_u = \min\{t, X_1, X_2, \ldots, X_K\}, \quad (15)$$

where $W_u = t$ denotes the case where the user is not updated in that update cycle. The ccdf of X_k is given by

$$\mathbb{P}(X_k > x) = \begin{cases} \frac{c_k u_k}{c_k - u_k}\left(\frac{e^{-u_k x}}{u_k} - \frac{e^{-c_k x}}{c_k}\right), & x \geq 0, \\ 1, & x < 0. \end{cases} \quad (16)$$

Since W_u takes only positive values, $\mathbb{E}[W_u]$ can be found by integrating its ccdf, i.e., $\mathbb{E}[W_u|T_s = t] = \int_0^\infty \mathbb{P}(W_u > x)dx$,

$$\mathbb{E}[W_u|T_s = t] = \int_0^t \mathbb{P}(X_1 > x)\cdots\mathbb{P}(X_K > x)dx. \quad (17)$$

For ease of exposition, let $p_v = (p_i)_{i\in[k]} \in \Pi_k\{c_k, u_k\} = V_p$, and $S_c = \sum_{k=1}^K \mathbb{1}\{p_k = c_k\}$. Then, $\mathbb{E}[W_u|T_s = t]$ equals

$$\frac{\prod_k c_k \prod_k u_k}{\prod_k (c_k - u_k)}\left(\sum_{p_v\in V_p} \frac{(-1)^{S_c}\left(1 - e^{-t(\sum_k p_k)}\right)}{(\sum_k p_k)\prod_k p_k}\right). \quad (18)$$

Next, we find $\mathbb{E}[W_u] = \mathbb{E}[\mathbb{E}[W_u|T_s]]$ as

$$\mathbb{E}[W_u] = \frac{\prod_k c_k \prod_k u_k}{\prod_k (c_k - u_k)}\left(\sum_{p_v\in V_p} \frac{(-1)^{S_c}}{\prod_k p_k}\frac{1}{\lambda + \sum_k p_k}\right). \quad (19)$$

Since $\mathbb{E}[T_u] = \mathbb{E}[T_s] - \mathbb{E}[W_u]$, we find $F_u(i)$ in (9) as

$$F_u(i) = 1 - \frac{\prod_k c_k \prod_k u_k}{\prod_k (c_k - u_k)}\left(\sum_{p_v\in V_p} \frac{(-1)^{S_c}}{\prod_k p_k}\frac{1}{1 + \frac{\sum_k p_k}{\lambda}}\right). \quad (20)$$

We note that when $K = 1$, i.e., single-cache system, the user freshness in (20) reduces to the expression in (14). When $K = 2$, i.e., two-cache system, the user freshness in (20) reduces to the expression in (21) at the top of the next page. Interestingly, comparing (14) and (21), we note that freshness in a two-cache system with update rates (c_1, c_2) from the source to the caches and (u_1, u_2) from caches to the user, yields a smaller freshness than in a single-cache system with an update rate $c = c_1 + c_2$ from the source to a cache and $u = u_1 + u_2$ from the cache to the user due to the negative term in (21).

III. Structure of the Optimal Policy

In this section, we find the optimum update rate allocation structure for general K and N. First, we consider the system with $K = 2$ caches and $N = 2$ files. We denote route k as the file update path from source through cache k to the user. Again dropping file index i, let user update rates for file i be u_1 and u_2 in route 1 and route 2, respectively, also let cache update rates in route 1 and route 2 be c_1 and c_2, respectively. We define the average variables as $\bar{u} = \frac{u_1+u_2}{2}$ and $\bar{c} = \frac{c_1+c_2}{2}$, and deviation from the average as $b = \frac{u_2-u_1}{2}$ and $a = \frac{c_2-c_1}{2}$. Thus, $u_1 = \bar{u} - b$, $u_2 = \bar{u} + b$, $c_1 = \bar{c} - a$, and $c_2 = \bar{c} + a$.

In the next lemma, for given user rates u_1 and u_2 (therefore, given $\bar{u}$ and b), and the total cache rate $2\bar{c}$, we find the optimal distribution of cache rates to maximize the freshness at the user, that is, we find the optimal a, a^*, in terms of b, $\bar{u}$ and $\bar{c}$.

Lemma 1 *In a cache update system with $K = 2$ parallel caches and $N = 2$ files, for given user rates u_1 and u_2, and the total cache rate $2\bar{c}$, the optimal cache rates are equal to $c_1^* = \bar{c} - a^*$ and $c_2^* = \bar{c} + a^*$ where*

$$a^* = \min\left\{b + \frac{(\bar{c} + \lambda + \bar{u})}{b(2\bar{c} + \lambda)}\left(\bar{u}(2\bar{c} + \lambda + \bar{u}) - b^2 - \sqrt{(\bar{u}^2 - b^2)((2\bar{c} + \lambda + \bar{u})^2 - b^2)}\right), \bar{c}\right\}. \quad (23)$$

Proof: We prove the lemma by writing (21) equivalently as (22) after inserting $\bar{u}$, $\bar{c}$, a and b. Since $\bar{u}$ and $\bar{c}$ are fixed, the first term and pre-factor of the second term in (22) are fixed. Taking the derivative of the term inside the parentheses with respect to a yields the first part of the min in (23). As this critical point yields $\frac{\partial^2 F_u(i)}{\partial a^2} < 0$, we conclude that a^* in (23) maximizes the freshness at the user. We note that $\frac{\partial a^*}{\partial b} \geq 0$, and thus, a^* increases monotonically with b, till it reaches $\bar{c}$, after which a^* is equal to $\bar{c}$, yielding (23). ■

$$F_u(i) = \frac{(u_1+u_2)(c_1+c_2)}{(\lambda+u_1+u_2)(\lambda+c_1+c_2)} - \frac{\lambda}{(\lambda+u_1+u_2)(\lambda+c_1+c_2)}\left(\frac{u_2c_1}{(\lambda+u_1+c_2)}+\frac{u_1c_2}{(\lambda+u_2+c_1)}\right) \quad (21)$$

$$F_u(i) = \frac{4\bar{c}\bar{u}}{(\lambda+2\bar{c})(\lambda+2\bar{u})} - \frac{\lambda}{(\lambda+2\bar{c})(\lambda+2\bar{u})}\left(\frac{(\bar{u}-b)(\bar{c}+a)}{(\bar{c}+\lambda+\bar{u}+b-a)}+\frac{(\bar{u}+b)(\bar{c}-a)}{(\bar{c}+\lambda+\bar{u}+a-b)}\right) \quad (22)$$

As an aside, we remark that in (23) we have $a^* \geq b$ as long as $a^* < \bar{c}$, that is, for a deviation b of u_1, u_2 from their average $\bar{u}$, the optimal a yields a bigger deviation for c_1^*, c_2^* from their average $\bar{c}$.

Next, we define $\tilde{F}_u(i)$ as the *cache-update-rate-optimized* freshness, where for fixed u_1, u_2, we insert the optimal cache update rates c_1^* and c_2^* in (21). Note that $\tilde{F}_u(i)$ is a function of b, $\bar{u}$ and $\bar{c}$. In the following lemma, we show that as u_1, u_2 get more *lopsided*, i.e., as the difference (u_2-u_1) increases, cache-update-rate-optimized freshness $\tilde{F}_u(i)$ increases.

Lemma 2 *$\tilde{F}_u(i)$ is an increasing function of b.*

We prove Lemma 2 by showing $\frac{d\tilde{F}_u(i)}{db} > 0$. Lemma 2 implies that lopsided update rates at the user increase the freshness.

Next, for a $K=2$ cache system with $N=2$ files, we show that we should restrict at least one of the files to a single route, that is, lopside at least one of the files to an extreme.

Lemma 3 *In a cache update system with $K=2$ caches and $N=2$ files, in the optimal policy, we need to restrict at least one file to a single route.*

Proof: Let the average rates at the caches and at the user hold values $\bar{c}_i = \frac{c_{1i}+c_{2i}}{2}$ and $\bar{u}_i = \frac{u_{1i}+u_{2i}}{2}$ for $i=1,2$ which fixes total user rates and total cache rates. Similarly, we have $u_{1i} = \bar{u}_i - b_i$ and $u_{2i} = \bar{u}_i + b_i$ which satisfies the total update rate constraints $u_{11}+u_{12}=U_1$ and $u_{21}+u_{22}=U_2$. Then, we change the update rates at the user to $u'_{11} = \bar{u}_1 - b_1 - \delta_1$, $u'_{21} = \bar{u}_1 + b_1 + \delta_1$, $u'_{12} = \bar{u}_2 - b_2 + \delta_2$ and $u'_{22} = \bar{u}_2 + b_2 - \delta_2$ such that we have $|\delta_1| = |\delta_2|$, $u'_{11}+u'_{12}=U_1$, and $u'_{21}+u'_{22}=U_2$ still hold. We analyze two cases of shuffling, shown in Fig. 3.

In the first case, increasing b_i for one file leads to increasing b_i value for the other file as shown in Fig. 3(a). As distributions of user rates for both files become lopsided simultaneously, it is a win-win situation for both files. For this case, we increase b_i values of files till one file is completely in a single route. For example, in Fig. 3(a), the user rates for the second file (shown in yellow) are $\bar{u}_2 - b_2$ and $\bar{u}_2 + b_2$ in route 1 and route 2, respectively. Then, we increase b_2 till $\bar{u}_2 - b_2 = 0$ in route 1 and the second file is completely restricted to route 2. Such shuffling also leads to a simultaneous increase in b_1.

In the second case, increasing b_i value of one file decreases b_i value of the other file. This case is shown in Fig. 3(b) where both files have larger user update rates in route 2. In order to determine which file to prioritize, we compare $\frac{d\tilde{F}_u(i)}{db_i}$ for both files. If $\frac{d\tilde{F}_u(1)}{db_1} > \frac{d\tilde{F}_u(2)}{db_2}$, then we prioritize improving freshness of file 1. One can show that $\frac{d^2\tilde{F}_u(i)}{db_i^2} > 0$. Thus, the increase in freshness of file 1 is always larger than the decrease in freshness of file 2. Similarly, if $\frac{d\tilde{F}_u(2)}{db_2} > \frac{d\tilde{F}_u(1)}{db_1}$, then we increase the freshness of the second file which decreases the freshness of the first file. Thus, we need to restrict at least one file to a single route to obtain the optimum freshness. ■

Thus, for a $K=2$ cache and $N=2$ file system with a given set of update rates u_{11}, u_{21}, u_{12} and u_{22}, we can shuffle these rates to increase the total freshness while keeping average rates $\bar{u}_1$, $\bar{u}_2$, $\bar{c}_1$, and $\bar{c}_2$ the same. In this process, we always end up restricting one of the files to only one route. Extending this result to a $K=2$ cache but arbitrary N files case, we iteratively choose a pair of files and increase freshness of the pair by restricting one of these files to a single route. We repeat this process until we restrict $N-1$ files to a single route each. Thus, for a $K=2$ cache, arbitrary N file system, only at most one file will be updated through both relays, and the remaining $N-1$ files will settle to a single relay.

Lemma 4 *Freshness of a file in a K-cache system with update rates at the cache $(c_1, c_2, c_3, \ldots, c_K)$, and at the user $(u_1, u_2, u_3, \ldots, u_K)$ is smaller than the freshness in a $(K-1)$-cache system with update rates at the cache $(c_1+c_2, c_3, \ldots, c_K)$, and at the user $(u_1+u_2, u_3, \ldots, u_K)$.*

Proof: With notation of Section II, since freshness $F_u(i) = 1 - \lambda E[W_u]$, where $E[W_u] = \int_0^\infty E[W_u|T_s=t]\lambda e^{-\lambda t}dt$, we prove the lemma by showing $E[W_u^K|T_s=t] - E[W_u^{K-1}|T_s=t] \geq 0$, where K in W_u^K denotes K-cache system. ■

Thus, given total update rates $\sum_{k=1}^K u_{ki}$ and $\sum_{k=1}^K c_{ki}$ for a file i, the maximum freshness is obtained by concentrating the rates in a single route to the extent possible. In the next section, we provide an approximate way of finding total update rates for files and scheduling them to individual links.

IV. Approximate Solution

The freshness maximization problem for our system is,

$$\begin{aligned} \max_{c_{ki},u_{ki}} \quad & \sum_{i=1}^N F_u(i) \\ \text{s.t.} \quad & \sum_{k=1}^K \sum_{i=1}^N c_{ki} \leq C \\ & \sum_{i=1}^N u_{ki} \leq U_k, \quad k=1,\ldots,K, \\ & c_{ki} \geq 0,\ u_{ki} \geq 0,\ k=1,\ldots,K,\ i=1,\ldots,N. \end{aligned} \quad (24)$$

This parallel-cache problem is significantly more complex than the cascade-cache problem in [42]. A Lagrangian approach as in [42] seems prohibitive as it results in highly nonlinear KKT

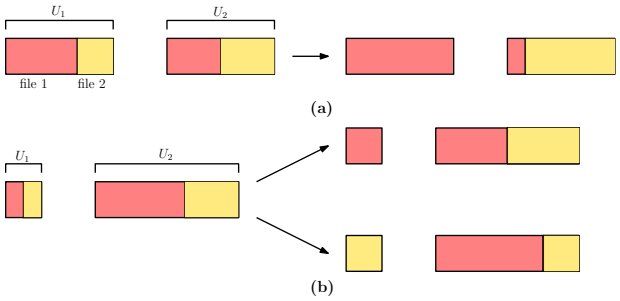


Fig. 3. Shuffling user rates for improving freshness. (a) Freshness of both files improve (file 2 only in route 2). (b) In upper branch, freshness of file 1 decreases and of file 2 increases (file 2 only in route 2). In lower branch, freshness of file 1 increases and of file 2 decreases (file 1 only in route 2).

conditions. We pursue an approximate solution approach utilizing the properties of the optimal solution found in Section 3.

First, we construct a single-cache problem by bringing all relays together, where the source-to-cache total update constraint is C and the relay-to-user total update constraint is $U = \sum_{k=1}^{K} U_k$. The optimal solution of this single-cache problem forms an upper bound for the optimum solution of our multi-cache problem, as it allows distributed relays to share update rate capacities. We denote this upper bound by F_{ub}.

Second, we extract a feasible solution for our multi-cache problem from the optimum solution of the constructed single-cache problem. We know from Lemma 4 that files need to be restricted to single routes for maximum freshness. Thus, our approximate solution takes the optimum solution of the constructed single-cache problem, and distributes the update rates in the multi-cache setting in such a way that each file is updated only through a single relay to the extent possible. Let the solution of the single-cache problem be u_i which is $u_i = \sum_{k=1}^{K} u_{ki}$. We assign the files in order of decreasing u_i to one of the routes. We start with the first route and fit fully as many files as possible, till we reach a file which will not fit completely and we make it split rates with the last route (route K). We follow this for $K-1$ routes. If a file rate u_i exceeds route capacity U_k, we first fill maximal full routes with it, then try to fit the remaining rate fully in the remaining routes. This leaves us with at most $K-1$ files that split rates between two routes. The remaining files go to route K. This approximate solution gives us a sub-optimal freshness F_{so}. Denoting the optimal freshness in our problem in (24) as F^*, we have

$$F_{so} < F^* < F_{ub} \tag{25}$$

which means $F^* - F_{so} \leq F_{ub} - F_{so}$, i.e., the gap between the sub-optimal solution and the optimal solution is bounded by the gap between the upper bound and the sub-optimal solution.

Next, we bound $F_{ub} - F_{so}$. We note that, in the sub-optimal policy, we assign at most $K-1$ files to two routes. From Lemma 2, freshness for file i increases when b_i increases, with minimum at $b_i = 0$ ($a^* = 0$) and maximum at $b_i = \bar{u}_i$ ($a^* = \bar{c}$). Hence, using (22), we find an upper bound on maximum freshness loss ratio ρ possible for a file due to splitting,

$$\rho = \frac{F_u(i)|_{(b_i,a_i^*)=(\bar{u}_i,\bar{c}_i)} - F_u(i)|_{(b_i,a_i^*)=(0,0)}}{F_u(i)|_{(b_i,a_i^*)=(\bar{u}_i,\bar{c}_i)}}, \tag{26}$$

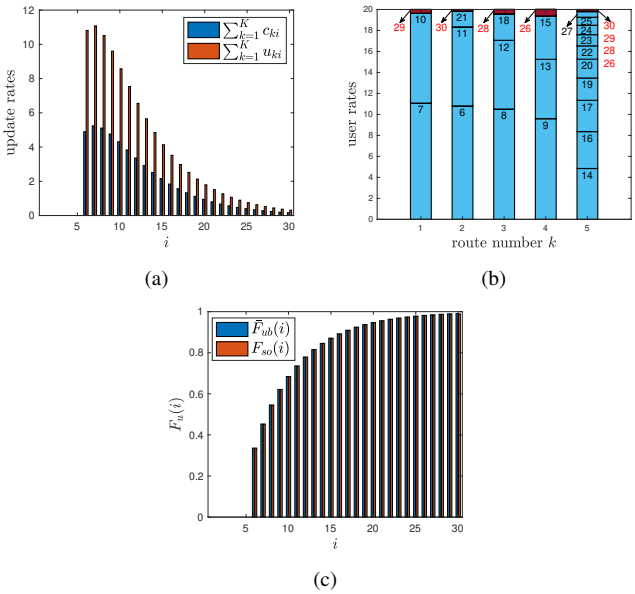


Fig. 4. (a) Total user rates and total cache rates obtained from the auxiliary solution, (b) route allocations for files (files with blue rates in single routes), and (c) freshness obtained for the single-cache and parallel cache systems.

which is equal to $\rho = \frac{\lambda_i}{2(\lambda_i+\bar{u}_i+\bar{c}_i)} < 0.5$. Since $F_{ub}(i) < 1$, the optimality gap is $F^* - F_{so} \leq \rho(K-1)F_{ub}(i) < 0.5(K-1)$. These $K-1$ files have low u_is, owing to very high or very low λ_is, as observed in [42]. In the former case, $F_{ub}(i)$ is low, while in the latter case, ρ is very low.

V. Numerical Results

We choose number of routes $K = 5$, number of files $N = 30$, total update rate at source $C = 50$ and at caches $U = 100$ where each route has $U_k = 20$. We use update arrival rates $\lambda_i = bq^i$ at the source for $i = 1, \ldots, N$, where $b > 0$, $q = 0.7$, and $\sum_{i=1}^{N} \lambda_i = a$, with $a = 100$. Note that the update arrival rates at the source λ_i decrease with the file index since $q < 1$.

We apply alternating maximization approach described in [42] to solve the auxiliary single-cache problem to obtain total update rate for file i at the user $\sum_{k=1}^{K} u_{ki}$ and at the caches $\sum_{k=1}^{K} c_{ki}$ as shown in Fig. 4(a). The total cache update rate constraint, i.e., $\sum_{k=1}^{K}\sum_{i=1}^{N} c_{ki} \leq C$, is already satisfied by both problems. In a parallel cache system, each route has its own total update rate constraint $\sum_{i=1}^{N} u_{ki} \leq U_k$, whereas the single-cache system has only one total update rate constraint for the user, i.e., $\sum_{k=1}^{K}\sum_{i=1}^{N} u_{ki} \leq U$ where $U = \sum_{k=1}^{K} U_k$. Thus, we need to choose the user rate allocation for all files in each route as described in Section IV, with corresponding cache rates found by (23) which are shown in Fig. 4(b). We denote the freshness at the user for the single-cache system obtained by the method in [42] as $\bar{F}_{ub}$. We plot $\bar{F}_{ub}$ and F_{so} in Fig. 4(c). Even though we split the update rates among different routes for $K - 1 = 4$ files that have some of the highest freshness (files 26, 28, 29, 30), their freshness loss is negligible as shown in Fig. 4(c). In this system, the total freshness loss, i.e., $\bar{F}_{ub} - F_{so}$, is equal to 0.0026, which is much smaller than the theoretical upper bound $0.5(K-1) = 2$.

REFERENCES

[1] E. Najm, R. D. Yates, and E. Soljanin. Status updates through M/G/1/1 queues with HARQ. In *IEEE ISIT*, June 2017.

[2] A. Soysal and S. Ulukus. Age of information in G/G/1/1 systems. In *Asilomar Conference*, November 2019.

[3] J. Cho and H. Garcia-Molina. Effective page refresh policies for web crawlers. In *ACM Transactions on Database Systems*, volume 28, pages 390–426, December 2003.

[4] A. Kolobov, Y. Peres, E. Lubetzky, and E. Horvitz. Optimal freshness crawl under politeness constraints. In *ACM SIGIR Conference*, July 2019.

[5] S. Farazi, A. G. Klein, and D. R. Brown III. Average age of information for status update systems with an energy harvesting server. In *IEEE Infocom*, April 2018.

[6] X. Wu, J. Yang, and J. Wu. Optimal status update for age of information minimization with an energy harvesting source. In *IEEE Transactions on Green Communications and Networking*, volume 2, pages 193–204, March 2018.

[7] A. Baknina, O. Ozel, J. Yang, S. Ulukus, and A. Yener. Sending information through status updates. In *IEEE ISIT*, June 2018.

[8] S. Leng and A. Yener. Age of information minimization for an energy harvesting cognitive radio. In *IEEE Transactions on Cognitive Communications and Networking*, volume 5, page 427–43, June 2019.

[9] A. Arafa, J. Yang, S. Ulukus, and H. V. Poor. Age-minimal transmission for energy harvesting sensors with finite batteries: Online policies. In *IEEE Transactions on Information Theory*, volume 66, pages 534–556, January 2020.

[10] M. A. Abd-Elmagid, H. S. Dhillon, and N. Pappas. A reinforcement learning framework for optimizing age of information in RF-powered communication systems. In *IEEE Transactions on Communications*, volume 68, pages 4747–4760, May 2020.

[11] E. T. Ceran, D. Gunduz, and A. Gyorgy. A reinforcement learning approach to age of information in multi-user networks. In *IEEE PIMRC*, September 2018.

[12] J. Liu, X. Wang, and H. Dai. Age-optimal trajectory planning for UAV-assisted data collection. In *IEEE Infocom*, April 2018.

[13] M. A. Abd-Elmagid and H. S. Dhillon. Average peak age-of-information minimization in UAV-assisted IoT networks. In *IEEE Transactions on Vehicular Technology*, volume 68, pages 2003–2008, February 2019.

[14] M. Bastopcu and S. Ulukus. Minimizing age of information with soft updates. In *Journal of Communications and Networks*, volume 21, pages 233–243, June 2019.

[15] M. Bastopcu and S. Ulukus. Timely group updating. In *CISS*, March 2021.

[16] R. D. Yates and S. K. Kaul. The age of information: Real-time status updating by multiple sources. In *IEEE Transactions on Information Theory*, volume 65, pages 1807–1827, March 2019.

[17] I. Kadota, A. Sinha, E. Uysal-Biyikoglu, R. Singh, and E. Modiano. Scheduling policies for minimizing age of information in broadcast wireless networks. In *IEEE/ACM Transactions on Networking*, volume 26, pages 2637–2650, December 2018.

[18] Y. Hsu. Age of information: Whittle index for scheduling stochastic arrivals. In *IEEE ISIT*, June 2018.

[19] B. Buyukates, A. Soysal, and S. Ulukus. Age of information scaling in large networks with hierarchical cooperation. In *IEEE Globecom*, December 2019.

[20] A. M. Bedewy, Y. Sun, S. Kompella, and N. B. Shroff. Age-optimal sampling and transmission scheduling in multi-source systems. In *ACM MobiHoc*, July 2019.

[21] B. Buyukates, A. Soysal, and S. Ulukus. Age of information in multihop multicast networks. In *Journal of Communications and Networks*, volume 21, pages 256–267, July 2019.

[22] M. Wang, W. Chen, and A. Ephremides. Reconstruction of counting process in real-time: The freshness of information through queues. In *IEEE ICC*, July 2019.

[23] M. Bastopcu and S. Ulukus. Who should Google Scholar update more often? In *IEEE Infocom*, July 2020.

[24] Y. Sun, Y. Polyanskiy, and E. Uysal-Biyikoglu. Remote estimation of the Wiener process over a channel with random delay. In *IEEE ISIT*, June 2017.

[25] M. Bastopcu and S. Ulukus. Timely tracking of infection status of individuals in a population. In *IEEE Infocom*, May 2021.

[26] J. Yun, C. Joo, and A. Eryilmaz. Optimal real-time monitoring of an information source under communication costs. In *IEEE CDC*, December 2018.

[27] C. Kam, S. Kompella, and A. Ephremides. Age of incorrect information for remote estimation of a binary Markov source. In *IEEE Infocom*, July 2020.

[28] J. Chakravorty and A. Mahajan. Remote estimation over a packet-drop channel with Markovian state. In *IEEE Transactions on Automatic Control*, volume 65, pages 2016–2031, July 2020.

[29] P. Mayekar, P. Parag, and H. Tyagi. Optimal source codes for timely updates. In *IEEE Transactions on Information Theory*, volume 66, pages 3714–3731, March 2020.

[30] M. Bastopcu, B. Buyukates, and S. Ulukus. Selective encoding policies for maximizing information freshness. *IEEE Transactions on Communications*, pages 1–1, February 2021.

[31] D. Ramirez, E. Erkip, and H. V. Poor. Age of information with finite horizon and partial updates. In *IEEE ICASSP*, May 2020.

[32] B. Buyukates and S. Ulukus. Timely distributed computation with stragglers. In *IEEE Transactions on Communications*, volume 68, pages 5273–5282, September 2020.

[33] P. Zou, O. Ozel, and S. Subramaniam. Optimizing information freshness through computation–transmission tradeoff and queue management in edge computing. *IEEE/ACM Transactions on Networking*, 29(2):949–963, February 2021.

[34] E. Ozfatura, B. Buyukates, D. Gunduz, and S. Ulukus. Age-based coded computation for bias reduction in distributed learning. In *IEEE Global Communications Conference*, Taipei, Taiwan, December 2020.

[35] H. H. Yang, A. Arafa, T. Q. S. Quek, and H. V. Poor. Age-based scheduling policy for federated learning in mobile edge networks. In *IEEE ICASSP*, May 2020.

[36] N. Rajaraman, R. Vaze, and R. Goonwanth. Not just age but age and quality of information. *IEEE Journal on Selected Areas in Communications*, 39(5):1325–1338, March 2021.

[37] M. Bastopcu and S. Ulukus. Age of information for updates with distortion. In *IEEE ITW*, August 2019.

[38] O. Ayan, M. Vilgelm, M. Klügel, S. Hirche, and W. Kellerer. Age-of-information vs. value-of-information scheduling for cellular networked control systems. In *ACM ICCPS*, April 2019.

[39] S. Banerjee, R. Bhattacharjee, and A. Sinha. Fundamental limits of age-of-information in stationary and non-stationary environments. In *IEEE ISIT*, June 2020.

[40] R. D. Yates, P. Ciblat, A. Yener, and M. Wigger. Age-optimal constrained cache updating. In *IEEE ISIT*, June 2017.

[41] H. Tang, P. Ciblat, J. Wang, M. Wigger, and R. D. Yates. Age of information aware cache updating with file- and age-dependent update durations. In *18th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOPT)*, June 2020.

[42] M. Bastopcu and S. Ulukus. Information freshness in cache updating systems. *IEEE Transactions on Wireless Communications*, 20(3):1861–1874, March 2021.

[43] M. Bastopcu and S. Ulukus. Maximizing information freshness in caching systems with limited cache storage capacity. In *Asilomar Conference*, November 2020.

[44] S. Zhang, J. Li, H. Luo, J. Gao, L. Zhao, and X. S. Shen. Towards fresh and low-latency content delivery in vehicular networks: An edge caching aspect. In *IEEE WCSP*, October 2018.

[45] W. Gao, G. Cao, M. Srivatsa, and A. Iyengar. Distributed maintenance of cache freshness in opportunistic mobile networks. In *IEEE ICDCS*, June 2012.

[46] N. Pappas, Z. Chen, and M. Hatami. Average AoI of cached status updates for a process monitored by an energy harvesting sensor. In *CISS*, March 2020.

[47] C. Kam, S. Kompella, G. D. Nguyen, J. Wieselthier, and A. Ephremides. Information freshness and popularity in mobile caching. In *IEEE ISIT*, June 2017.

[48] Y. Gu, Q. Wang, H. Chen, Y. Li, and B. Vucetic. Optimizing information freshness in two-hop status update systems under a resource constraint. *IEEE Journal on Selected Areas in Communications*, 39(5):1380–1392, March 2021.

[49] A. Arafa and S. Ulukus. Timely updates in energy harvesting two-hop networks: Offline and online policies. *IEEE Transactions on Wireless Communications*, 18(8):4017–4030, August 2019.