Article

Cached Files Updating Revisited: The Distribution of Popularity-Weighted Average File Age

Jixiang Zhang

[1] School of Information Science and Engineering, Southeast University; 230179077@seu.edu.cn
* Correspondence: 230179077@seu.edu.cn

Abstract: In this paper, using the discrete time model, we consider the average age of all files for a cached-files-updating system where a server generates N files and transmits them to a local cache. In order that the cached files are fresh, in each time slot the server updates files with certain probabilities. The age of one file or its age of information (AoI) is defined as the time the file stays in cache since it was last time sent to cache. Assume that each file in cache has corresponding request popularity. In this paper, we obtain the distribution function of the popularity-weighted average age over all files, which gives a complete description of this average age. For the random age of single file, both the mean and its distribution have been derived before by establishing a simple Markov chain. Using the same idea, we show that an N dimensional stochastic process can be constituted to characterize the changes of N file ages simultaneously. By solving the steady-state of the resulting process, we obtain the explicit expression of stationary probability for an arbitrary state-vector. Then, the distribution function of the popularity-weighted average age can be derived by mergering a proper set of stationary probabilities. For the possible applications, the distribution function can be utilized to calculate the probability that the average age violates certain statistical guarantee.

Keywords: age of information; cached files updating; stationary distribution; discrete time model

1. Introduction

The cached files updating system consists of one server and a local cache. The server generates N files and transmits them to the local cache. The users can only request files from the cache because the server is assumed to be far away. The server has to refresh the stored files from time to time in order that the cached files are fresh enough. The freshness of one file is measured by its age, or the age of information (AoI) which is defined as the time this file stays at the cache since the last time it was delivered to the cache. The AoI is a newly proposed metric and is widely used to characterize the timeliness of an information transmission system. Since the AoI was introduced, great progress has been made and lots of research results have been published in recent years. An introduction and survey of the AoI was given in a recent paper [1], in which the authors summarize the recent contributions in the broad area of the AoI.

In general, some files in cache are more popular but some others are not. Assume that each file has different request popularity, then the popularity-weighted average age of all the files can be defined. The cached files updating model arises in lots of settings. For example, in an online game system with a central server, the files in cache denote the players in the game system. Some users interact with the server more often, while others interact less. In addition, every player has a user level and the server tends to update the status of those players who have higher levels. Observing that the communication frequency of a player in the game system corresponds to the file popularity in cache system. One user's level affects how often the server interacts with this user, which corresponds to the updating probability of the file in the cached files updating system.

 doi:10.20944/preprints202009.0113.v1

Refreshing the cached files in real time was first formulated in [2]. In paper [2], a remote server generates multiple different files and transmits them to the local cache, so that the users can request files. The authors assume that each file has its own request popularity and define the average age of all cached files according to the file popularities. Then, it was asked how the server should update files such that users can receive the most recent version of their requests. By solving a relaxed optimization problem, it was proved that an asymptotically optimal policy should update each file in proportion to the square root of its popularity. Other variants concerning minimum-AoI cached files updating include [3–5]. The authors in [3] considered the case where several sources generate updating packets and deliver them to a local server. In every source, assume that the successive packet arrivals form a Poisson process, and is independent of other arriving processes. They analyzed the average AoI of one user's requests during a period of time, assuming that all the source-states are updated periodically. In addition, another related metric called age of synchronization (AoS) is also discussed. The results obtained in [3] suggested that the optimal updating frequency of each source should depend only on the square root of the source popularity, which is similar to the conclusion in work [2]. In a recent paper [4], the refreshing frequency of each file is supposed to be a function of its instantaneous AoI. For this case, the average age of all the files is optimized over all the feasible schemes that the server can use. Furthermore, in work [5] the cache size is supposed to be limited. Notice that in this case the cache cannot store all the files and part of the user demands may fail. Therefore, in paper [5] the user was allowed to download files directly from the server with a higher download cost, compared with requesting the files from the local cache. Then, the average AoI and the total download costs are jointly optimized over all the possible updating schemes applied at the server.

Under the discrete time model, many system design problems based on AoI are considered in recently years, such as [6–16]. A variety of optimization methods are used even combining with the machine learning [17–23]. Many average or peak AoI expressions are proposed in terms of various measures such as the length of file, the download cost, the download energy consumed for each file, and so on. Since the majority of these problems are solved using optimization theory or machine learning methods, the analytical structures of the resulting solutions are often not clear. We found that the analysis of the AoI for the general status updating system in discrete time model was not definitely proposed until an recent paper [24]. The discrete time updating system was also analyzed in another work [25]. Only simple queue models are considered in [24], such as Ber/G/1 and G/G/1. Meanwhile, only average AoI was discussed and does not mention the distribution of the steady state AoI. The methods used in [24] mainly come from the previous works, which considered the AoI in continuous time model. The stationary distribution of the AoI was discussed in [25]. The article [25] is a little bit complicated, the authors intend to describe a sample path of the discrete AoI process and relate this process with a queueing system. For the discrete time queues they considered, the transmission time of each packet is one time slot. We also use this assumption in the current paper. On the other hand, the approach used in [25] is hard to generalize if we want to analyze more general status updating models. For example, for the case there are two or more parallel servers in the updating system, the method of [25] may be useless.

The analysis of the discrete time status updating system is just beginning. Obtaining the distribution function of the steady state AoI can be very hard in the continuous time setting. For the cached files updating model, in this paper we show that the distribution function of the popularity-weighted average age can be determined under the discrete time model. Thus, we give the complete description of the popularity-weighted average age. Define a vector which is composed of all the file ages, we call it the age-vector or the state-vector. We constitute an N dimensional discrete stochastic process, which describes the changes of all file ages from one time slot to the next time slot. Then, the stationary probability of an arbitrary age-state can be found if we solve the so-called stationary equations of the resulting discrete process. As long as all the stationary probabolities are obtained, the distribution of the average file age defined according to an arbitrary file's request popularity can be determined by mergering a proper group of stationary probabilities.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 September 2020 doi:10.20944/preprints202009.0113.v1

The rest of this paper is organized as follows. The system model is given in section 2, the popularity-weighted average age over all the cached files is also defined. In section 3, we solve the simple case where only two files are generated at the server. Then, in section 4, the methods used for the simple example are applied for the general cases. Firstly, we derive the explicit stationary probabilities for the case $N = 3$ by establishing a three-dimensional stochastic process and solving its steady state. Aften then the general case is handled where an N-dimensional random process is constituted. Furthermore, some more general models of cached files updating are also mentioned. For example, we consider the case where multiple-file-updating is allowed in each time slot, and give the stationary equations of the stochastic process established for this case. In the end, we conclude this paper in section 5.

2. System Model and Problem Formulation

The server generates N files and sends them to a local cache. Assume that the files from the server are always new. That is, the ages of all the files maintained in server are equal to zero. The user can only download the files from the local cache because the server is located far away. The files stored in cache are becoming obsolete over time. Thus, in order that the files are new enough, the server has to refresh the cached files from time to time. A basic cached files updating model is depicted in Figure 1.

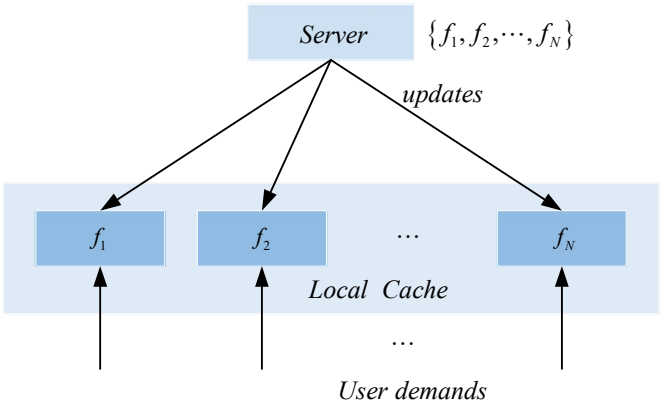

Figure 1. System model for the cached files updating.

In the discrete time model, the time is discretized into multiple time slots. At each time slot, the server updates one file randomly according to an i.i.d. distribution. Assume that the files have different request popularities. We want to determine the distribution function of the popularity-weighted average age over all the cached files. One file's popularity is defined as the percentage the requests for this file accounts for over all user requests in a long period of time. Usually, the file's request popularity is represented by a probability distribution.

Notice that in current paper, we assume that transmitting each file from the server to cache consumes exactly one time slot.

Definition 1. *Denote $\overline{\Delta}$ as the time average AoI over all cached files according to a group of file popularities $\{p_i, 1 \leq i \leq N\}$, then $\overline{\Delta}$ is defined as*

$$\overline{\Delta} = \sum_{i=1}^{N} p_i \lim_{T\to\infty} \frac{1}{T} \sum_{k=1}^{T} a_i(k) \tag{1}$$

where $a_i(k)$ represents the age of the ith file at the kth time slot.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 September 2020 doi:10.20944/preprints202009.0113.v1

Definition 2. *Denote $a_1, a_2, \ldots, a_N$ as the random ages of files $f_1, f_2, \ldots, f_N$, respectively. The popularity-weighted average file age Δ_{all} defined according to a set of file popularities $p_i, 1 \leq i \leq N$ is*

$$\Delta_{all} = \sum_{i=1}^{N} p_i a_i$$

which is also a random variable.

The ergodicity property is then used, which is assumed to hold in the AoI literatures. Ergodicity ensures that the time average AoI defined in (1) converges to its statistical average as time goes to infinity.We have

$$\overline{\Delta} = \mathbb{E}[\Delta_{all}]$$

3. The Mean of Popularity-Weighted Average Age and Its Distribution for the Case $N = 2$

In this section, we first give the mean of the popularity-weighted average file age. The distribution function is then computed for the simple case $N = 2$. In addition, we also explain why the size of the cache cannot be limited when the average file age is analyzed.

Assume that at the beginning of each time slot, the server selects to refresh the file $f_i, 1 \leq i \leq N$ according to an i.i.d. distribution $\{c_i, 1 \leq i \leq N\}$. The refreshing distribution is supposed to be independent of the file's popularity. Let the age of file f_i be the random variable $a_i, 1 \leq i \leq N$.

We declare that all the a_i's are independent. It is observed that a_i depends only on its value at the previous one time slot and the file independently updated by the server at the current time slot. As a result, a_i has nothing to do with $a_j, j \in \{1, 2, \ldots, N\} \setminus \{i\}$. Therefore, the average age of all cached files Δ_{all} equals the sum of N independent random variables $p_i a_i, 1 \leq i \leq N$.

At every time slot, because the server decides to update file f_i with probability c_i independently, the successive updates of f_i form a Bernoulli process with parameter c_i, which is equivalent to say the time interval between two updates follows a geometric distribution.

It has been obtained before the average age of single file with a geometrical updating interval is equal to $1/c_i$, if the updating probability at each time slot is c_i. The stationary distribution of the single file's age is proved to be a geometric distribution with parameter c_i. For the paper to be self-contained, we restate these results in the following.

Assume that at each time slot the server refreshes the file f with probability c, such that the age of f jumps to 1. Otherwise, its age increases 1. Define the file age as the state, we show that the state transtions can be described by Figure 2.

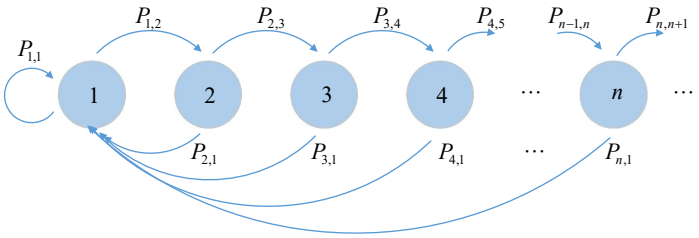

Figure 2. An example of age-state transfers for the single file.

Since in every time slot the server updates the file f independently and identically, all the forward transfers from n to $(n+1)$ occur with probability $(1-c)$. For any state n, with probability c it changes

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 September 2020 doi:10.20944/preprints202009.0113.v1

to 1 at the next time slot. Denote the stationary probability of n as π_n. When the random process reaches the steady state, the stationary equations are written as

$$\begin{cases} \pi_n = \pi_{n-1}(1-c) & (n \geq 2) \\ \pi_1 = \sum_{k=1}^{\infty} \pi_k c \end{cases}$$

Because the sum of all the stationary probabilities has to be 1, it is easy to solve that

$$\pi_n = c(1-c)^{n-1} \qquad (n \geq 1)$$

which shows that the age of f is geometrically distributed in steady state. The mean of the random age is calculated as $\sum_{n=1}^{\infty} n \cdot \pi_n = 1/c$.

Hence, for the mean of Δ_{all}, we have

$$\overline{\Delta} = \mathbb{E}[\Delta_{all}] = \mathbb{E}\left[\sum_{i=1}^{N} p_i a_i\right] = \sum_{i=1}^{N} p_i \mathbb{E}[a_i] = \sum_{i=1}^{N} \frac{p_i}{c_i} \tag{2}$$

We summarize the above results as a Theorem.

Theorem 1. *Assume that the server updates files according to a distribution $\{c_i, 1 \leq i \leq N\}$, which is independent of the file popularities $\{p_i, 1 \leq i \leq N\}$. Then, the random file ages $a_1, a_2, \ldots, a_N$ are geometric random variables and are independent of each other. The successive arriving updates for each file form an independent Bernoulli process. The mean of the popularity-weighted average age $\overline{\Delta}$ is given as*

$$\overline{\Delta} = \mathbb{E}[\Delta_{all}] = \sum_{i=1}^{N} \frac{p_i}{c_i}$$

where the popularity-weighted average file age Δ_{all} is defined as

$$\Delta_{all} = \sum_{i=1}^{N} p_i a_i$$

Equation (2) shows that for the mean age $\overline{\Delta}$ gets smaller, the larger refreshing probabilities should be assigned to those files who have greater request popularities. This result can be proved by using the following simple inequality.

Lemma 1. *(Rearrangement Inequality [26]) Assume that there are two sets of number $\{A_i, 1 \leq i \leq M\}$ and $\{B_i, 1 \leq i \leq M\}$ satisfying $A_1 \leq A_2 \leq \cdots \leq A_M$, $B_1 \leq B_2 \leq \cdots \leq B_M$. Then, we have*

$$\sum_{i=1}^{M} A_i B_{M+1-i} \leq \sum_{i=1}^{M} A_i C_i \leq \sum_{i=1}^{M} A_i B_i$$

where $\{C_i, 1 \leq i \leq M\}$ is an arbitrary permutation of $\{B_i, 1 \leq i \leq M\}$.

Next, the distribution of the average age Δ_{all} is considered. Observing that Δ_{all} is the sum of N independent random variables $p_i a_i, 1 \leq i \leq N$, then its distribution can be determined as the convolution of N respective distributions.

$$\Pr\{\Delta_{all} = j\} = \Pr\left\{\sum_{i=1}^{N} p_i a_i = j\right\} = P_{a_1} \otimes P_{a_2} \otimes \cdots \otimes P_{a_N} \tag{3}$$

where we use $P_{a_i}, 1 \leq i \leq N$ to represent the stationary distribution of $p_i a_i$, which is also geometric since we have known all the a_i's are geometrically distributed.

So far, the mean of the average age $\overline{\Delta}$ is obtained in (2), and in equation (3) even the probability distribution of Δ_{all} is determined as the convolutions of N distributions $P_{a_i}, 1 \leq i \leq N$. However, expression (3) is implicit and more importantly, the method above may be useless when the more

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 September 2020 doi:10.20944/preprints202009.0113.v1

general cached-files-updating models are considered. For example, it is possible that no file or multiple files are updated in each time slot. In addition, the refreshing probability of a file may be time-varying and be related to its request popularity. The method we used above may be ineffective due to the reason the independence of a_i, $1 \leq i \leq N$ does not hold or the distributions of a_i's are hard to obtain. Instead, in the following we show that by establishing a discrete stochastic process, the average age analysis can be solved. Even for the more general system models, the random changes of file ages can also be described by a properly constituted random process. In this section, we determine the distribution function of Δ_{all} for the case $N = 2$, while the general cases where $N \geq 3$ are discussed in section 4.

By the way, we show that for the case the size of the cache is limited, the popularity-weighted average age cannot be defined. There are some confusions if some files are not in the cache. First of all, assume that the cache is full and a file f comes. Let the coming file does not contained in cache previously. The cache has to decide whether the file f is accepted or not. In order to accommodate such new files, some other files originally stored in cache must be discarded. When a possible deleting-and-updating scheme is considered at the cache, some trade-offs between several factors are often caused.

Skipping over this first problem and putting aside the user requests, observing that the random age of single file can still be analyzed, even when this file has been deleted from the cache at some time. Notice that the age of f increases by one constantly at each time slot as long as it is not updated by the server, which is independent of whether or not the file f is in cache. When an update of f is received, assume that with a probability the cache discards one another file and accepts this update. Since it consumes one time slot from the server to cache, so that the new file has age 1. In this way, we show that the age analysis for single file can be carried on. However, another problem occurs if the "average" file age is considered. Because the set of files stored in cache change with time, the sum age averaged over the cached files cannot be well defined unless all the files are homogeneous. Further, the average age of all N files in the limited-size-cache case cannot be defined as well, because we do not know how large the age of f is if the file f is not contained in cache.

Taking the file request popularities into consideration will make the age analysis more complex. Notice that when the user downloads the files from the cache, it is possible that some requests may fail since the corresponding files are not contained in cache at that time. Thus, certain schemes must be added, such that the user requests are satisfied. It is meaningful to discuss the freshness of cached files only when the user obtains all demand files successfully. If some files are not stored in cache at all times, then using the local cache only the updating system cannot ensure that the user gets the requested files. In such cases the model is defective, the analysis of average file age is meaningless.

Now, we compute the distribution function of Δ_{all} for the case $N = 2$.

Assume that the server generates only two different files and the cache size equals two as well. Define a two-dimensional integer vector $(a_{1,k}, a_{2,k})$ where $a_{1,k}$, $a_{2,k}$ denote the ages of the files f_1 and f_2 at the kth time slot. Establishing the stochastic process $Age_2 = \{(a_{1,k}, a_{2,k}), k \geq 1\}$. Since at every time slot one of two files are updated by the server, so that either $a_{1,k}$ or $a_{2,k}$ equals 1 at all times. Let the initial state is $(1,1)$ and denote the stationary probability for the state (n_1, n_2) as $\pi_{(n_1,n_2)}$, we show that the stationary equations for the process Age_2 are determined as

$$\begin{cases} \pi_{(n,1)} = \pi_{(n-1,1)} c_2 & (n \geq 3) \\ \pi_{(2,1)} = \left(\sum_{k=1}^{\infty} \pi_{(1,k)}\right) c_2 & \\ \pi_{(1,n)} = \pi_{(1,n-1)} c_1 & (n \geq 3) \\ \pi_{(1,2)} = \left(\sum_{k=1}^{\infty} \pi_{(k,1)}\right) c_1 & \end{cases} \tag{4}$$

Consider the case $n \geq 3$, the state $(n,1)$ can only be obtained from $(n-1,1)$ and let the server update f_2. Notice that one of two parameters have to be 1 at each time slot, it is observed that $(n_1, 1)$ cannot be obtained from a state of form $(1, n_2)$. Therefore, the first line of (4) holds. Similarly, the third

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 September 2020 doi:10.20944/preprints202009.0113.v1

line of (4) can also be obtained when the state considered is $(1,n)$. On the other hand, lots of states can transfer to $(1,2)$ or $(2,1)$ because when a file is updated, its age can take multiple values.

Solving the system of equations (4) is not hard. For the case $n \geq 2$, it shows that

$$\pi_{(n,1)} = \pi_{(n-1,1)}c_2 = \cdots = \pi_{(2,1)}c_2^{n-2} = \left(\sum_{k=2}^{\infty} \pi_{(1,k)}\right) c_2^{n-1} \tag{5}$$

$$\pi_{(1,n)} = \pi_{(1,n-1)}c_1 = \cdots = \pi_{(1,2)}c_1^{n-2} = \left(\sum_{k=2}^{\infty} \pi_{(k,1)}\right) c_1^{n-1} \tag{6}$$

Summing up both sides of (5) and (6) from $n = 2$ to infinity and rearranging the terms we obtain that

$$\sum_{n=2}^{\infty} \pi_{(n,1)} = \left(\sum_{k=2}^{\infty} \pi_{(1,k)}\right) \frac{c_2}{c_1}$$

$$\sum_{n=2}^{\infty} \pi_{(1,n)} = \left(\sum_{k=2}^{\infty} \pi_{(k,1)}\right) \frac{c_1}{c_2}$$

Observing that the two file ages never return to $(1,1)$ after it departs from $(1,1)$, because at each time slot only one file can be refreshed. Therefore, we obtain that $\pi_{(1,1)}$ is equal to 0. Since all the probabilities must add up to 1, we have the relation

$$1 = \sum_{n=2}^{\infty} \pi_{(1,n)} + \sum_{n=2}^{\infty} \pi_{(n,1)}$$

from which we obtain

$$\sum_{n=2}^{\infty} \pi_{(n,1)} = c_2, \quad \sum_{n=2}^{\infty} \pi_{(1,n)} = c_1 \tag{7}$$

Thus, the stationary probabilities for all the state vectors are determined as

$$\pi_{(n,1)} = c_1 c_2^{n-1}, \quad \pi_{(1,n)} = c_2 c_1^{n-1} \quad (n \geq 2) \tag{8}$$

Next, we compute the distribution function $\Pr\{\Delta_{all} = j\}$, assuming that the two file popularities are p_1 and p_2. Observing that j can take any value of form $p_1 + p_2 n_2$ or $p_1 n_1 + p_2$ where $n_1, n_2 \in \mathbb{N}$.

$$\begin{aligned} \Pr\{\Delta_{all} = j\} &= \Pr\{p_1 a_1 + p_2 a_2 = j\} \\ &= \pi_{(1,\frac{j-p_1}{p_2})} + \pi_{(\frac{j-p_2}{p_1},1)} \\ &= c_2 c_1^{\frac{j-p_1}{p_2}-1} + c_1 c_2^{\frac{j-p_2}{p_1}-1} \end{aligned} \tag{9}$$

On the other hand, using the convolution formula (3) the distribution of Δ_{all} can also be determined for the simple case $N = 2$.

$$\begin{aligned} \Pr\{\Delta_{all} = j\} &= \Pr\{p_1 a_1 + p_2 a_2 = j\} \\ &= \Pr\{p_1 + p_2 a_2 = j\} + \Pr\{p_1 a_1 + p_2 = j\} \qquad (10) \\ &= \Pr\{a_2 = \frac{j-p_1}{p_2}\} + \Pr\{a_1 = \frac{j-p_2}{p_1}\} \\ &= c_2(1-c_2)^{\frac{j-p_1}{p_2}-1} + c_1(1-c_1)^{\frac{j-p_2}{p_1}-1} \\ &= c_2 c_1^{\frac{j-p_1}{p_2}-1} + c_1 c_2^{\frac{j-p_2}{p_1}-1} \qquad (11) \end{aligned}$$

where in (10) we use the observation that one of the two file ages must equal 1 and the relation $c_1 + c_2 = 1$ is applied in equation (11).

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 September 2020 doi:10.20944/preprints202009.0113.v1

Remember that we have mentioned before for the single file f, its age is geometrically distributed. The parameter equals the corresponding updating probability. Therefore, the two respective age distributions are

$$\Pr\{a_i = l\} = c_i(1-c_i)^{l-1} \qquad (i \in \{1,2\}, l \geq 1)$$

4. Distribution Function of Popularity-Weighted Average Age for General Cases

In this section, the general case where N files are stored in cache is considered. We obtain the distribution of Δ_{all} by constituting a N-dimensional stochastic process

$$Age_N = \{(a_{1,k}, a_{2,k}, \ldots, a_{N,k}), k \geq 1\}$$

At the beginning, assume that all the cached files have age 1. First of all, for the age-vectors in the state space of Age_N, we describe their characteristics in a proposition. After then, the stationary equations of the process Age_N are given in Theorem 2. The remainder of this section is then divided into three subsections. In subsection 4.1, we solve the stationary equations for the case $N = 3$ and determine the explicit expression of stationary probability for each state vector in Theorem 3. The idea that solving the case $N = 3$ can be developed to handle the general case, where N files are stored in cache. In part 4.2, we obtain the stationary probability for an arbitrary N dimensional state vector in Theorem 4. Finally, several more general system models are discussed in subsection 4.3. In particular, assume that the server can update multiple files in each time slot, a random process is constituted and its stationary equations are given in Theorem 5.

Proposition 1. *For the state space of the random process $Age_N = \{(a_{1,k}, a_{2,k}, \ldots, a_{N,k}), k \geq 1\}$, the following statements hold:*

(i) except for the initial state, exactly there is one vector component equal to 1 for an arbitrary state vector;

(ii) if the files f_i and f_j, $i, j \in \{1, 2, \ldots, N\}$ are updated by the server at least one time, then they will never have the same age, i.e., $a_i \neq a_j$ at all times;

(iii) the only case where several files have the same age is when all of them are not refreshed by the server from start to finish. In addition, the value of this same age is maximal over all the components of the state vector.

Moreover, for the states with several identical components, their stationary probabilities equal to zero. As a result, when solving the stationary equations, we only need to consider those states where all the files have different ages.

Proof. The first statement is obvious. Because at each time slot the server randomly updates one file, so that one of the N file ages are reset to 1. Thus, at all times there is one component equal to 1 in an arbitrary state vector. Assume that f_i and f_j are updated at two different time slots, then their ages a_i and a_j will not be equal because two ages start at different times, and both start from 1. This proves the statement (ii). Finally, since we assume that the initial state is $(1, \ldots, 1)$, if some files are not updated from start to finish, then their ages will still be identical. Observing that updating a file can only decrease its age, then the files having not been refreshed from start to end will have the maximal age. This proves the result (iii).

Next, we state the proof of the remaining part of the Proposition. Suppose that the state vector at the kth time slot is $(1, n_2, n_3, \ldots, n_i, M, \ldots, M)$, where the last $(N-i)$ file ages are all equal to M. Due to the statement (iii), all of them are never updated by the server so that their ages are maximal. Without loss of generality, for the first k components of the state vector, assume that $1 < n_2 < n_3 < \cdots < n_i$. By constantly finding the state at the previous one time slot, we can obtain

$$(1, 2, n_3 - n_2 + 2, \ldots, n_i - n_2 + 2, M - n_2 + 2, \ldots, M - n_2 + 2)$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 September 2020 doi:10.20944/preprints202009.0113.v1

Table 1. States backward from $(1,3,5,7,7)$

State	Time slot	Assumption
$(1,3,5,7,7)$	k	
$(1,2,4,6,6)$	$k-1$	
$(l_1,1,3,5,5)$	$k-2$	$l_1=4$
$(3,1,2,4,4)$	$k-3$	
$(2,l_2,1,3,3)$	$k-4$	$l_2=2$
$(1,1,l_3,2,2)$	$k-5$	$l_3=2$
$(1,1,2,2,2)$		

Table 2. States backward from $(1,3,5,7,7)$ to $(1,1,1,1,1)$

State	Time slot	Let l_i be maximal
$(1,3,5,7,7)$	k	
$(1,2,4,6,6)$	$k-1$	
$(l_1,1,3,5,5)$	$k-2$	$l_1=5$
$(4,1,2,4,4)$	$k-3$	
$(3,l_2,1,3,3)$	$k-4$	$l_2=3$
$(2,2,1,2,2)$	$k-5$	
$(1,1,1,1,1)$	$k-6$	

at the $(k-n_2+2)$th time slot. Go backward one time slot further, it was observed that the first file f_1 was updated. Assume that the age of f_1 before this refreshing is denoted by l_1, we represent the age vector for all the files in the $(k-n_2+1)$th time slot as

$$(l_1,1,n_3-n_2+1,\ldots,n_i-n_2+1,M-n_2+1,\ldots,M-n_2+1)$$

Because the last $(N-i)$ files have the largest age over all the files, we show that the relation $l_1 \leq M-n_2+1$ holds. Continue with this procedure, assume that $l_1 > n_i-n_2+1$ such that the age of f_3 returns to 2 at the $(k-n_3+2)$th time slot. Then, in the $(k-n_3+1)$th time slot, it was shown that f_2 was updated by the server. Denote the state vector as

$$(l_1-n_3+n_2,l_2,1,\ldots,n_i-n_3+1,M-n_3+1,\ldots,M-n_3+1)$$

Also, we have $l_2 \leq M-n_3+1$ hold and assume that $l_2 > n_i-n_3+1$. Proceeding in this manner, eventually it was shown that the state returns to a vector containing multiple "1". For instance, starting with $(1,3,5,7,7)$, we give an example of the state evolutions in Table 1.

Specially, if the l_1,l_2 and l_3 in Table 1 equal the maximal values they can take, we show that the state vector would return to $(1,1,1,1,1)$ in the end. We give the state evolutions in Table 2.

For the first case, we reduce the state vector to a state with multiple "1". However, the statement (i) tells that there is only one file having age 1 in an arbitrary state vector. Therefore, these states are impossible and their stationary probabilities equal zero. In Table 2, we give a procedure that reducing an age-vector with identical components to initial state $(1,\ldots,1)$. This implies that the state is achievable from the initial state. But notice that the stationary probability of $(1,\ldots,1)$ is itself equal to zero, since the state vector will never transfers to $(1,\ldots,1)$ again as long as it jumps out of $(1,\ldots,1)$. Combining both cases, we conclude that for the state vectors with several identical components, their stationary probabilities are all equal to zero. In other words, for all the cases multiple files having the same age, we show that the stationary probabilities are all zero. This completes the proof of the Proposition. □

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 September 2020 doi:10.20944/preprints202009.0113.v1

After the proof of Proposition, in the following we determine the stationary equations of the stochastic process Age_N.

Define $\pi_{(n_1,n_2,\ldots,n_N)}$ as the stationary probability of the state $(n_1, n_2, \ldots, n_N)$. Due to the Proposition above, we only need to consider the state vectors where all the file ages are different.

First of all, consider the state vector $(1, n_1, n_2, \ldots, n_{N-1})$ where all the ages are different and $n_k \geq 3$, $1 \leq k \leq N-1$. It is easy to show that the probability $\pi_{(1,n_1,\ldots,n_{N-1})}$ is equal to $\pi_{(1,n_1-1,\ldots,n_{N-1}-1)}c_1$. Notice that at any time exactly one of the file ages equal 1. Here, the file must be f_1, since we assume that the ages of other $(N-1)$ files are all greater than 3. In this case, at the previous one time slot, the state vector can only be $(1, n_1-1, \ldots, n_{N-1}-1)$.

Generally, for the state $(n_1, \ldots, n_{j-1}, 1, n_{j+1}, \ldots, n_N)$ where $n_j = 1$, the similar results can also be obtained. Let $n_k \geq 3, k \in \{1, 2, \ldots, N\} \setminus \{j\}$, we show that

$$\pi_{(n_1,\ldots,n_{j-1},1,n_{j+1},\ldots,n_N)} = \pi_{(n_1-1,\ldots,n_{j-1}-1,1,n_{j+1}-1,\ldots,n_N-1)}c_j \tag{12}$$

Next, assume that the age of f_1 equals 1 and let $n_j = 2$ for the jth file f_j. At the previous one time slot, the server must update f_1, so that its age jumps to 1. Notice that the age of f_1 before this update can take lots of values. The probability $\pi_{(1,\ldots,n_{j-1},2,n_{j+1},\ldots,n_N)}$ is equal to

$$\pi_{(1,\ldots,n_{j-1},2,n_{j+1},\ldots,n_N)} = \left(\sum\nolimits_{\text{all proper } k} \pi_{(k,n_2-1,\ldots,n_{j-1}-1,1,n_{j+1}-1,\ldots,n_N-1)}\right) c_1 \tag{13}$$

$$= \left(\sum\nolimits_{k=2}^{\infty} \pi_{(k,n_2-1,\ldots,n_{j-1}-1,1,n_{j+1}-1,\ldots,n_N-1)}\right) c_1 \tag{14}$$

The age k of f_1 in equation (13) can take any values as long as it is different from all other file ages. Actually, simply summing up all the stationary probabilities from $k = 2$ to ∞ is also feasible, because the Proposition shows that the stationary probability is zero if the corresponding state vector has identical components.

A total of $N(N-1)$ equations like (14) can be obtained by considering all the cases $n_i = 1$ and $n_j = 2$ where $i, j \in \{1, 2, \ldots, N\}$.

We state the results about the stationary equations in Theorem 2 below.

Theorem 2. *Assume that all the file ages are different, the stationary equations for the stochastic process $Age_N = \{(a_{1,k}, a_{2,k}, \ldots, a_{N,k}), k \geq 1\}$ are determined as*

$$\pi_{(n_1,\ldots,n_{i-1},1,n_{i+1},\ldots,n_N)} = \pi_{(n_1-1,\ldots,n_{i-1}-1,1,n_{i+1}-1,\ldots,n_N-1)}c_i \qquad n_l \geq 3, l \in \{1, 2, \ldots, N\} \setminus \{i\} \tag{15}$$

$$\pi_{(1,\ldots,n_{j-1},2,n_{j+1},\ldots,n_N)} = \left(\sum\nolimits_{k=2}^{\infty} \pi_{(k,\ldots,n_{j-1}-1,1,n_{j+1}-1,\ldots,n_N-1)}\right) c_1 \quad n_l \geq 3, l \in \{1, 2, \ldots, N\} \setminus \{1, j\} \tag{16}$$

Notice that all the permutations of state $(n_1, n_2, \ldots, n_N)$ should be included. We have different equations like (15) and (16) corresponding to every permutation of $(n_1, n_2, \ldots, n_N)$.

4.1. Obtaining the stationary probabilities for the case $N = 3$

In the following, we derive all the stationary probabilities by solving the system of equations in Theorem 2. First of all, for the case $N = 3$, we compute the explicit expression of the stationary probability for each state vector. The results are given in Theorem 3.

Theorem 3. *Consider the case where the server generates three different files. Assume that at each time slot the server updates the file f_i with probability c_i, $i \in \{1, 2, 3\}$. The solution to the stationary equations, or the stationary probabilities for all the state vectors are determined by*

$$\pi_{(1,2,3)} = \pi_{(1,3,2)} = \pi_{(2,1,3)} = \pi_{(2,3,1)} = \pi_{(3,1,2)} = \pi_{(3,2,1)} = c_1c_2c_3 \tag{17}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 September 2020 doi:10.20944/preprints202009.0113.v1

$$\begin{cases} \pi_{(1,2,n)} = \pi_{(2,1,n)} = c_1c_2c_3(c_1+c_2)^{n-3} & (n \geq 3) \\ \pi_{(1,n,2)} = \pi_{(2,n,1)} = c_1c_2c_3(c_1+c_3)^{n-3} & (n \geq 3) \\ \pi_{(n,1,2)} = \pi_{(n,2,1)} = c_1c_2c_3(c_2+c_3)^{n-3} & (n \geq 3) \end{cases} \tag{18}$$

The stationary probabilies for the states $(1, n_1, n_2)$ *where* $n_1, n_2 \geq 3$ *are equal to*

$$\pi_{(1,n_1,n_2)} = \begin{cases} c_1c_2c_3(c_1+c_2)^{n_2-3}\left(\frac{c_1}{c_1+c_2}\right)^{n_1-2} & (n_2 > n_1 \geq 3) \\ c_1c_2c_3(c_1+c_3)^{n_1-3}\left(\frac{c_1}{c_1+c_3}\right)^{n_2-2} & (n_1 > n_2 \geq 3) \end{cases} \tag{19}$$

For the states $(n_1, 1, n_2)$ *we show that* $\pi_{(n_1,1,n_2)}$ *equals*

$$\pi_{(n_1,1,n_2)} = \begin{cases} c_1c_2c_3(c_1+c_2)^{n_2-3}\left(\frac{c_2}{c_1+c_2}\right)^{n_1-2} & (n_2 > n_1 \geq 3) \\ c_1c_2c_3(c_1+c_3)^{n_1-3}\left(\frac{c_2}{c_2+c_3}\right)^{n_2-2} & (n_1 > n_2 \geq 3) \end{cases} \tag{20}$$

and the probabilities $\pi_{(n_1,n_2,1)}$ *are determined by*

$$\pi_{(n_1,n_2,1)} = \begin{cases} c_1c_2c_3(c_1+c_3)^{n_2-3}\left(\frac{c_3}{c_1+c_3}\right)^{n_1-2} & (n_2 > n_1 \geq 3) \\ c_1c_2c_3(c_2+c_3)^{n_1-3}\left(\frac{c_3}{c_2+c_3}\right)^{n_2-2} & (n_1 > n_2 \geq 3) \end{cases} \tag{21}$$

Proof. Observing that from an arbitrary state $(n_1, n_2, 1)$, if the server first updates f_2 and then updates f_1, the state vector will finally change to $(1,2,3)$ with the middle state $(n_1+1, 1, 2)$. Thus, we have the equation

$$\pi_{(1,2,3)} = \left[\left(\sum_{n_1,n_2} \pi_{(n_1,n_2,1)}\right) c_2\right] c_1$$

For the other permutations of the state $(1,2,3)$, by the same idea, we have

$$\begin{aligned} \pi_{(1,3,2)} &= \left[\left(\sum_{n_1,n_2} \pi_{(n_1,1,n_2)}\right) c_3\right] c_1, & \pi_{(2,1,3)} &= \left[\left(\sum_{n_1,n_2} \pi_{(n_1,n_2,1)}\right) c_1\right] c_2 \\ \pi_{(3,1,2)} &= \left[\left(\sum_{n_1,n_2} \pi_{(1,n_1,n_2)}\right) c_3\right] c_2, & \pi_{(2,3,1)} &= \left[\left(\sum_{n_1,n_2} \pi_{(n_1,1,n_2)}\right) c_1\right] c_3 \\ & & \pi_{(3,2,1)} &= \left[\left(\sum_{n_1,n_2} \pi_{(1,n_1,n_2)}\right) c_2\right] c_3 \end{aligned}$$

Next, the three infinite sums are calculated so that all the probabilities in (17) can be obtained.

$$\begin{aligned} & \sum_{n_1,n_2} \pi_{(1,n_1,n_2)} \\ &= \sum_{n_1<n_2} \pi_{(1,n_1,n_2)} + \sum_{n_1>n_2} \pi_{(1,n_1,n_2)} \\ &= \sum_{j=1}^{\infty}\sum_{k=2}^{\infty} \pi_{(1,k,k+j)} + \sum_{j=1}^{\infty}\sum_{k=2}^{\infty} \pi_{(1,k+j,k)} \\ &= \sum_{j=1}^{\infty}\left(\pi_{(1,2,2+j)} + \pi_{(1,3,3+j)} + \pi_{(1,4,4+j)} + \dots\right) \\ &\qquad + \sum_{j=1}^{\infty}\left(\pi_{(1,2+j,2)} + \pi_{(1,3+j,3)} + \pi_{(1,4+j,4)} + \dots\right) \\ &= \sum_{j=1}^{\infty}\left(\pi_{(1,2,2+j)} + \pi_{(1,2,2+j)}c_1 + \pi_{(1,2,2+j)}c_1^2 + \dots\right) \\ &\qquad + \sum_{j=1}^{\infty}\left(\pi_{(1,2+j,2)} + \pi_{(1,2+j,2)}c_1 + \pi_{(1,2+j,2)}c_1^2 + \dots\right) \\ &= \frac{1}{1-c_1}\left(\sum_{j=1}^{\infty} \pi_{(1,2,2+j)} + \sum_{j=1}^{\infty} \pi_{(1,2+j,2)}\right) \end{aligned} \tag{22}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 September 2020 doi:10.20944/preprints202009.0113.v1

The two summations in (22) are dealt with further as follows.

$$\begin{aligned}
&\textstyle\sum_{j=1}^{\infty}\pi_{(1,2,2+j)}+\sum_{j=1}^{\infty}\pi_{(1,2+j,2)}\\
=&\left(\pi_{(1,2,3)}+\pi_{(1,2,4)}+\pi_{(1,2,5)}+\dots\right)+\left(\pi_{(1,3,2)}+\pi_{(1,4,2)}+\pi_{(1,5,2)}+\dots\right)\\
=&\left[\left(\textstyle\sum_{k=2}^{\infty}\pi_{(k,1,2)}\right)c_1+\left(\sum_{k=2}^{\infty}\pi_{(k,1,3)}\right)c_1+\left(\sum_{k=2}^{\infty}\pi_{(k,1,4)}\right)c_1+\dots\right]\\
&\quad+\left[\left(\textstyle\sum_{k=2}^{\infty}\pi_{(k,2,1)}\right)c_1+\left(\sum_{k=2}^{\infty}\pi_{(k,3,1)}\right)c_1+\left(\sum_{k=2}^{\infty}\pi_{(k,4,1)}\right)c_1+\dots\right]\\
=&\left(\textstyle\sum_{n_1,n_2}\pi_{(n_1,1,n_2)}\right)c_1+\left(\sum_{n_1,n_2}\pi_{(n_1,n_2,1)}\right)c_1
\end{aligned}$$

Therefore, we obtain the following relation

$$\textstyle\sum_{n_1,n_2}\pi_{(1,n_1,n_2)}=\frac{c_1}{1-c_1}\left(\sum_{n_1,n_2}\pi_{(n_1,1,n_2)}+\sum_{n_1,n_2}\pi_{(n_1,n_2,1)}\right)\tag{23}$$

For the other two sums, the similar equations can also be deduced as follows.

$$\textstyle\sum_{n_1,n_2}\pi_{(n_1,1,n_2)}=\frac{c_2}{1-c_2}\left(\sum_{n_1,n_2}\pi_{(1,n_1,n_2)}+\sum_{n_1,n_2}\pi_{(n_1,n_2,1)}\right)\tag{24}$$

$$\textstyle\sum_{n_1,n_2}\pi_{(n_1,n_2,1)}=\frac{c_3}{1-c_3}\left(\sum_{n_1,n_2}\pi_{(1,n_1,n_2)}+\sum_{n_1,n_2}\pi_{(n_1,1,n_2)}\right)\tag{25}$$

Combining equations (23), (24), (25) and notice that

$$1=\textstyle\sum_{n_1,n_2}\pi_{(1,n_1,n_2)}+\sum_{n_1,n_2}\pi_{(n_1,1,n_2)}+\sum_{n_1,n_2}\pi_{(n_1,n_2,1)}$$

we solve that

$$\begin{cases}\sum_{n_1,n_2}\pi_{(1,n_1,n_2)}=c_1\\ \sum_{n_1,n_2}\pi_{(n_1,1,n_2)}=c_2\\ \sum_{n_1,n_2}\pi_{(n_1,n_2,1)}=c_3\end{cases}\tag{26}$$

By equation (26) $\pi_{(1,2,3)}$ and the other five probabilities in (17) can be determined.

$$\begin{cases}\pi_{(1,2,3)}=\pi_{(2,1,3)}=\left(\sum_{n_1,n_2}\pi_{(n_1,n_2,1)}\right)c_2c_1=c_1c_2c_3\\ \pi_{(1,3,2)}=\pi_{(2,3,1)}=\left(\sum_{n_1,n_2}\pi_{(n_1,1,n_2)}\right)c_1c_3=c_1c_2c_3\\ \pi_{(3,1,2)}=\pi_{(3,2,1)}=\left(\sum_{n_1,n_2}\pi_{(1,n_1,n_2)}\right)c_2c_3=c_1c_2c_3\end{cases}$$

Thus, we derive the first six probabilities in Theorem 3. Next, for the general state $(1,2,n)$ where n is greater than 3, we have following calculations:

$$\begin{aligned}
\pi_{(1,2,n)}&=\left(\textstyle\sum_{k=2}^{\infty}\pi_{(k,1,n-1)}\right)c_1\\
&=\left(\pi_{(2,1,n-1)}+\sum_{k=3}^{\infty}\pi_{(k,1,n-1)}\right)c_1\\
&=\left[\pi_{(2,1,n-1)}+\sum_{k=3}^{\infty}\pi_{(k-1,1,n-2)}c_2\right]c_1\\
&=\left[\pi_{(2,1,n-1)}+\left(\sum_{k=2}^{\infty}\pi_{(k,1,n-2)}\right)c_2\right]c_1
\end{aligned}\tag{27}$$

Notice that $\left(\sum_{k=2}^{\infty}\pi_{(k,1,n-2)}\right)c_1=\pi_{(1,2,n-1)}$, from which we can derive that

$$\textstyle\sum_{k=2}^{\infty}\pi_{(k,1,n-2)}=\frac{\pi_{(1,2,n-1)}}{c_1}\tag{28}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 September 2020 doi:10.20944/preprints202009.0113.v1

in equation (27). Substituting (28) into (27) yields the following recursive formula

$$\pi_{(1,2,n)} = \pi_{(2,1,n-1)}c_1 + \pi_{(1,2,n-1)}c_2 \tag{29}$$

Similarly, for the stationary probabilities $\pi_{(2,1,n)}$, $n \geq 3$ it shows that

$$\begin{aligned}
\pi_{(2,1,n)} &= \left(\sum_{k=2}^{\infty} \pi_{(1,k,n-1)}\right) c_2 \\
&= \left(\pi_{(1,2,n-1)} + \sum_{k=3}^{\infty} \pi_{(1,k,n-1)}\right) c_2 \\
&= \left[\pi_{(1,2,n-1)} + \sum_{k=3}^{\infty} \pi_{(1,k-1,n-2)}c_1\right] c_2 \\
&= \left[\pi_{(1,2,n-1)} + \left(\sum_{k=2}^{\infty} \pi_{(1,k,n-2)}\right) c_1\right] c_2 \\
&= \left[\pi_{(1,2,n-1)} + \frac{\pi_{(2,1,n-1)}}{c_2}c_1\right] c_2 \qquad (30) \\
&= \pi_{(1,2,n-1)}c_2 + \pi_{(2,1,n-1)}c_1 \qquad (31)
\end{aligned}$$

where in (30) the relation $\left(\sum_{k=2}^{\infty} \pi_{(1,k,n-2)}\right) c_2 = \pi_{(2,1,n-1)}$ is used. Observing the equations (29) and (31), we obtain that for $n \geq 3$,

$$\pi_{(1,2,n)} = \pi_{(2,1,n)} = \pi_{(1,2,n-1)}(c_1 + c_2) \tag{32}$$

Let $n \geq 3$, we directly give the recursive formulas for the state vectors $(1, n, 2)$ and $(n, 1, 2)$.

$$\begin{aligned}
\pi_{(1,n,2)} &= \pi_{(2,n,1)} = \pi_{(1,n-1,2)}(c_1 + c_3) \\
\pi_{(n,1,2)} &= \pi_{(n,2,1)} = \pi_{(n-1,1,2)}(c_2 + c_3)
\end{aligned}$$

Applying (32) iteratively yields

$$\pi_{(1,2,n)} = \pi_{(2,1,n)} = \pi_{(1,2,3)}(c_1 + c_2)^{n-3} = c_1c_2c_3(c_1 + c_2)^{n-3} \qquad (n \geq 3) \tag{33}$$

For the stationary probabilities $\pi_{(1,n,2)}$ and $\pi_{(n,1,2)}$, $n \geq 3$, using the same method the other two iterative equations can also be derived. We show that

$$\pi_{(1,n,2)} = \pi_{(2,n,1)} = \pi_{(1,3,2)}(c_1 + c_3)^{n-3} = c_1c_2c_3(c_1 + c_3)^{n-3} \qquad (n \geq 3) \tag{34}$$

and

$$\pi_{(n,1,2)} = \pi_{(n,2,1)} = \pi_{(3,2,1)}(c_2 + c_3)^{n-3} = c_1c_2c_3(c_2 + c_3)^{n-3} \qquad (n \geq 3) \tag{35}$$

Finally, for the case $n_1, n_2 \geq 3$ we determine the stationary probabilities for the most general states $(1, n_1, n_2)$, $(n_1, 1, n_2)$ and $(n_1, n_2, 1)$. Assume that $n_2 > n_1 \geq 3$ we have

$$\begin{aligned}
\pi_{(1,n_1,n_2)} &= \pi_{(1,2,n_2-n_1+2)}c_1^{n_1-2} = \pi_{(1,2,3)}(c_1 + c_2)^{n_2-n_1-1}c_1^{n_1-2} \\
&= c_1c_2c_3(c_1 + c_2)^{n_2-n_1-1}c_1^{n_1-2} = c_1c_2c_3(c_1 + c_2)^{n_2-3}\left(\frac{c_1}{c_1 + c_2}\right)^{n_1-2}
\end{aligned} \tag{36}$$

On the contrary, if $n_1 > n_2 \geq 3$ we see that $\pi_{(1,n_1,n_2)}$ can be computed as

$$\begin{aligned}
\pi_{(1,n_1,n_2)} &= \pi_{(1,n_1-n_2+2,2)}c_1^{n_2-2} = \pi_{(1,3,2)}(c_1 + c_3)^{n_1-n_2-1}c_1^{n_2-2} \\
&= c_1c_2c_3(c_1 + c_3)^{n_1-n_2-1}c_1^{n_2-2} = c_1c_2c_3(c_1 + c_3)^{n_1-3}\left(\frac{c_1}{c_1 + c_3}\right)^{n_2-2}
\end{aligned} \tag{37}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 September 2020 doi:10.20944/preprints202009.0113.v1

Combining equations (36) and (37) the stationary probability $\pi_{(1,n_1,n_2)}$ is given as

$$\pi_{(1,n_1,n_2)} = \begin{cases} c_1c_2c_3(c_1+c_2)^{n_2-3}\left(\frac{c_1}{c_1+c_2}\right)^{n_1-2} & (n_2 > n_1 \geq 3) \\ c_1c_2c_3(c_1+c_3)^{n_1-3}\left(\frac{c_1}{c_1+c_3}\right)^{n_2-2} & (n_1 > n_2 \geq 3) \end{cases} \tag{38}$$

The explicit expressions of $\pi_{(n_1,1,n_2)}$ and $\pi_{(n_1,n_2,1)}$ are directly given as follows. For $\pi_{(n_1,1,n_2)}$ we have

$$\pi_{(n_1,1,n_2)} = \begin{cases} c_1c_2c_3(c_1+c_2)^{n_2-3}\left(\frac{c_2}{c_1+c_2}\right)^{n_1-2} & (n_2 > n_1 \geq 3) \\ c_1c_2c_3(c_2+c_3)^{n_1-3}\left(\frac{c_2}{c_2+c_3}\right)^{n_2-2} & (n_1 > n_2 \geq 3) \end{cases}$$

and the probability $\pi_{(n_1,n_2,1)}$ is determined by

$$\pi_{(n_1,n_2,1)} = \begin{cases} c_1c_2c_3(c_1+c_3)^{n_2-3}\left(\frac{c_3}{c_1+c_3}\right)^{n_1-2} & (n_2 > n_1 \geq 3) \\ c_1c_2c_3(c_2+c_3)^{n_1-3}\left(\frac{c_3}{c_2+c_3}\right)^{n_2-2} & (n_1 > n_2 \geq 3) \end{cases}$$

So far, we complete the proof of Theorem 3. □

Provided all the stationary probabilities, in the following we determine the distribution function of the popularity-weighted average file age.

$$\begin{aligned} & \Pr\{\Delta_{all} = j\} \\ = \ & \Pr\{p_1a_1 + p_2a_2 + p_3a_3 = j\} \\ = \ & \Pr\{p_2a_2 + p_3a_3 = j - p_1\} + \Pr\{p_1a_1 + p_3a_3 = j - p_2\} + \Pr\{p_1a_1 + p_2a_2 = j - p_3\} \\ = \ & \textstyle\sum_{p_2n_1+p_3n_2=j-p_1} \pi_{(1,n_1,n_2)} + \sum_{p_1n_1+p_3n_2=j-p_2} \pi_{(n_1,1,n_2)} + \sum_{p_1n_1+p_2n_2=j-p_3} \pi_{(n_1,n_2,1)} \end{aligned}$$

Continue the calculation is tedious and not necessary. Since we have determined the stationary probability for every state vector (m_1, m_2, m_3), and the integer triples satisfying $p_1m_1 + p_2m_2 + p_3m_3 = j$ can be rapidly found by designing a simple computer program.

The solving method for the simple case $N = 3$ is also effective when we intend to find the explicit solutions to the stationary equations for the general case. Now, assume that N files are contained in cache. Firstly, the probability expression for a certain state vector is obtained. Then, we show that the other stationary probabilities can be derived by introducing a permutation operator and an induced one-to-one mapping. Without loss of generality, the probability $\pi_{(1,n_1,n_2,\ldots,n_{N-1})}$ is computed, assuming that the file ages satisfy $n_{N-1} > n_{N-2} > \cdots > n_2 > n_1$.

4.2. Solving the stationary equations when an arbitrary N files are contained in cache

We first state the Lemma 2, from which the stationary probability $\pi_{(1,n_1,n_2,\ldots,n_{N-1})}$ can easily be obtained.

Lemma 2. *The following statements concerning the stationary probabilities hold.*
(i) Let $\sigma(1,2,3,\ldots,N)$ be an arbitrary permutation of $(1,2,3,\ldots,N)$, then we have

$$\pi_{\sigma(1,2,3,\ldots,N)} = \pi_{(1,2,3,\ldots,N)} = \textstyle\prod_{i=1}^{N} c_i$$

(ii) Consider the state $(1,2,\ldots,k,n_{k+1},\ldots,n_N)$ where the first k components are the integers from 1 to k. It is shown that

$$\pi_{(1,2,\ldots,k,n_{k+1},\ldots,n_N)} = \pi_{(\sigma(1,2,\ldots,k),n_{k+1},\ldots,n_N)} \tag{39}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 September 2020 doi:10.20944/preprints202009.0113.v1

We use $(\sigma(1,2,\ldots,k), n_{k+1}, \ldots, n_N)$ *to represent the states where the latter* $(N-k)$ *components are fixed and the first k file ages can be any permutation of k-dimensional vector* $(1,2,\ldots,k)$.

(iii) For fixed k, $2 \le k \le N-1$ *we have the following recursive relation*

$$\pi_{(1,2,\ldots,k,n_{k+1},\ldots,n_N)} = \pi_{(1,2,\ldots,k,n_{k+1}-1,\ldots,n_N-1)} \left(\sum_{i=1}^{k} c_i\right) \tag{40}$$

Provided the above Lemma 2, we first prove the Theorem 4, in which we give the explicit expression of the stationary probability for an arbitrary state vector. After then, the Lemma 2 is proved.

Theorem 4. *For any state vector* $(m_1, m_2, \ldots, m_N)$ *where one of* m_i, $1 \le i \le N$ *equal 1. Let* σ *be the permutation operator satisfying* $\sigma(m_1, m_2, \ldots, m_N) = (n_0, n_1, \ldots, n_{N-1})$, *where* $1 = n_0 < n_1 < n_2 < \cdots < n_{N-1}$. *The permutation operator* σ *leads to a one-to-one mapping* $g_\sigma : \{1,2,3,\ldots,N\} \mapsto \{0,1,2,\ldots,N-1\}$ *which is determined by the following relationships*

$$m_k = n_{g_\sigma(k)}, \qquad k \in \{1,2,\ldots,N\}$$

Assuming that the update distribution of the server on each time slot is $\{c_i, 1 \le i \le N\}$, *we show that the stationary probability* $\pi_{(m_1,m_2,\ldots,m_N)}$ *is equal to*

$$\pi_{(m_1,m_2,\ldots,m_N)} = c_{g_\sigma^{-1}(N-1)} \prod_{j=1}^{N-1} c_{g_\sigma^{-1}(j-1)} \left(\sum_{k=1}^{j} c_{g_\sigma^{-1}(k-1)}\right)^{m_{g_\sigma^{-1}(j)} - m_{g_\sigma^{-1}(j-1)} - 1} \tag{41}$$

where $g_\sigma^{-1}(\cdot)$ *denotes the reverse mapping of* g_σ.

Proof. Because any state vector can always be converted to $(n_0, n_1, \ldots, n_{N-1})$ where the vector components satisfy $1 = n_0 < n_1 < n_2 < \cdots < n_{N-1}$ by a sequential of permutation operations, which we denote simply by σ. Therefore, in order to obtain the result for an arbitrary state vector, we only need to derive the stationary probabilities for those states whose components are arranged from smallest to largest. In other words, in order to prove Theorem 4 it suffices to compute $\pi_{(1,n_1,n_2,\ldots,n_{N-1})}$. We show that

$$\begin{aligned}
&\pi_{(1,n_1,n_2,n_3,\ldots,n_{N-1})} \\
&= \pi_{(1,2,n_2-n_1+2,n_3-n_1+2,\ldots,n_{N-1}-n_1+2)} c_1^{n_1-2} \\
&= \pi_{(1,2,3,n_3-n_2+3,\ldots,n_{N-1}-n_2+3)} (c_1+c_2)^{n_2-n_1-1} c_1^{n_1-2} && (42)\\
&= \pi_{(1,2,3,4,n_4-n_3+4,\ldots,n_{N-1}-n_3+4)} (c_1+c_2+c_3)^{n_3-n_2-1} (c_1+c_2)^{n_2-n_1-1} c_1^{n_1-2} && (43)\\
&\qquad\qquad \vdots \\
&= \pi_{(1,2,3,\ldots,N)} (c_1+c_2+\cdots+c_{N-1})^{n_{N-1}-n_{N-2}-1} \ldots (c_1+c_2)^{n_2-n_1-1} c_1^{n_1-2} && (44)\\
&= \left(\prod_{i=1}^{N} c_i\right) \prod_{j=1}^{N-1} \left(\sum_{k=1}^{j} c_k\right)^{n_j-n_{j-1}-1} && (45)\\
&= c_N \prod_{j=1}^{N-1} c_j \left(\sum_{k=1}^{j} c_k\right)^{n_j-n_{j-1}-1}
\end{aligned}$$

For the state $(1, n_1, n_2, \ldots, n_{N-1})$, notice that when all the n_i, $1 \le i \le N-1$ are greater than 3, the only feasible state vector at the previous one time slot is $(1, n_1-1, n_2-1, \ldots, n_{N-1}-1)$ and the first file f_1 must be updated. So, we can constantly decrease $n_i, 1 \le i \le N-1$ by one until n_1 becomes 2. From (42) to (44) the result (iii) in Lemma 2 is applied, for the case $k=2$, $k=3$ and $k=N-1$, respectively. The equation (45) holds due to the first result of Lemma 2, which shows that $\pi_{(1,2,3,\ldots,N)} = \prod_{i=1}^{N} c_i$. Therefore, we obtain the explicit expression of the stationary probability $\pi_{(1,n_1,n_2,n_3,\ldots,n_{N-1})}$.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 September 2020 doi:10.20944/preprints202009.0113.v1

In general, for an arbitrary state $(m_1, m_2, \ldots, m_N)$, we can find a permutation operator σ such that

$$\sigma(m_1, m_2, m_3, \ldots, m_N) = (n_0, n_1, n_2, \ldots, n_{N-1})$$

where $1 = n_0 < n_1 < n_2 < \cdots < n_{N-1}$. The N one-to-one corresponding relationships between $\{m_i, 1 \le i \le N\}$ and $\{n_j, 0 \le j \le N-1\}$ determine a one-to-one mapping

$$g_\sigma : \{1, 2, 3, \ldots, N\} \mapsto \{0, 1, 2, \ldots, N-1\}$$

which is defined by

$$m_k = n_{g_\sigma(k)} \qquad (1 \le k \le N)$$

Then, the kth updating probability c_k in the state vector $(n_0, n_1, n_2, \ldots, n_{N-1})$ corresponds to $c_{g_\sigma^{-1}(k-1)}$ in the original state $(m_1, m_2, m_3, \ldots, m_N)$ for $k = 1, 2, 3, \ldots, N$. Replacing c_k with $c_{g_\sigma^{-1}(k-1)}$ in (45) and observing that $n_j = m_{g_\sigma^{-1}(j)}$ we obtain that

$$\pi_{(m_1,m_2,m_3,\ldots,m_N)} = \left(\prod_{i=1}^{N} c_{g_\sigma^{-1}(i-1)}\right) \prod_{j=1}^{N-1} \left(\sum_{k=1}^{j} c_{g_\sigma^{-1}(k-1)}\right)^{m_{g_\sigma^{-1}(j)} - m_{g_\sigma^{-1}(j-1)} - 1} \tag{46}$$

$$= c_{g_\sigma^{-1}(N-1)} \prod_{j=1}^{N-1} c_{g_\sigma^{-1}(j-1)} \left(\sum_{k=1}^{j} c_{g_\sigma^{-1}(k-1)}\right)^{m_{g_\sigma^{-1}(j)} - m_{g_\sigma^{-1}(j-1)} - 1} \tag{47}$$

which gives the explicit formula of the stationary probability when all the file ages are represented by the state vector $(m_1, m_2, \ldots, m_N)$. Thus, we complete the proof of Theorem 4. □

At last, we state the proof of the Lemma 2.

The proof of Lemma 2. Observing that starting with an arbitrary state, if in the next N time slots every file is refreshed by the server exactly one time, then the ages of all the files after these N operations will be one of the permutations of $(1, 2, 3, \ldots, N)$. This fact shows that in the steady state the relation

$$\pi_{\sigma(1,2,3,\ldots,N)} = \pi_{(1,2,3,\ldots,N)} = \prod_{i=1}^{N} c_i$$

must hold, where $\sigma(1, 2, 3, \ldots, N)$ denotes an arbitrary permutation of state $(1, 2, 3, \ldots, N)$ and we obtain the first result of Lemma 2. The similar idea can be applied to the cases where some files are updated regardless of the others. Assume that the first k files are refreshed in some order by the server in consecutive k time slots, the k-dimensional sub-vector formed by the first k file ages must be some permutation of $(1, 2, 3, \ldots, k)$. Thus, we have the equation

$$\pi_{(\sigma(1,2,\ldots,k),n_{k+1},\ldots,n_N)} = \left(\sum \pi_{(*,*,\ldots,*,n_{k+1}-k,\ldots,n_N-k)}\right) \prod_{i=1}^{k} c_i \tag{48}$$

where the symbol "$*$" represents an arbitrary integer that can be used as the age of a cached file. Notice that the probability the server updates k files in turn is independent of the order these k operations are performed. Therefore, we prove the statement (ii) of Lemma 2, that is

$$\pi_{(\sigma(1,2,\ldots,k),n_{k+1},\ldots,n_N)} = \pi_{(1,2,\ldots,k,n_{k+1},\ldots,n_N)}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 September 2020 doi:10.20944/preprints202009.0113.v1

In the end, the recursive formula (40) is proved. We have the following equations

$$\pi_{(1,2,\ldots,k,n_{k+1},\ldots,n_N)} = \left(\sum_{j_1,j_2,\ldots,j_k} \pi_{(j_1,j_2,\ldots,j_k,n_{k+1}-k,\ldots,n_N-k)}\right) c_k c_{k-1}\ldots c_1 \tag{49}$$

$$\begin{aligned} &= \left(\sum_{j_2,j_3,\ldots,j_k} \pi_{(1,j_2,j_3,\ldots,j_k,n_{k+1}-k,\ldots,n_N-k)}\right) c_1 \prod_{i\neq 1} c_i \\ &\quad + \left(\sum_{j_1,j_3,\ldots,j_k} \pi_{(j_1,1,j_3,\ldots,j_k,n_{k+1}-k,\ldots,n_N-k)}\right) c_2 \prod_{i\neq 2} c_i \\ &\quad + \ldots \\ &\quad + \left(\sum_{j_1,j_2,\ldots,j_k} \pi_{(j_1,j_2,j_3,\ldots,1,n_{k+1}-k,\ldots,n_N-k)}\right) c_k \prod_{i\neq k} c_i \end{aligned} \tag{50}$$

$$= \sum_{l=1}^{k} \pi_{(\sigma_l(1,2,3,\ldots,k),n_{k+1}-1,\ldots,n_N-1)} c_l \tag{51}$$

$$= \pi_{(1,2,3,\ldots,k,n_{k+1}-1,\ldots,n_N-1)} \left(\sum_{l=1}^{k} c_l\right) \tag{52}$$

Starting with an arbitrary state vector of form $(j_1, j_2, j_3, \ldots, j_k, n_{k+1}-k, \ldots, n_N-k)$, by refreshing the first k files in the order from f_k to f_1, the state vector will jump to $(1, 2, \ldots, k, n_{k+1}, \ldots, n_N)$ in the end. Thus, the equation (49) holds. As previously mentioned in the Proposition, at any time there must be one file having age 1, thus the summation in (49) can be divided into k disjoint sums in (50) according to the file whose age is 1. For the equation (51), let the server update the first k files except the file of age 1 in an arbitrary order. The different order the server uses to update the other $(k-1)$ files will create different state vectors. However, notice that in (50) the latter $(N-k)$ components of those vectors are the same and the first k file ages are always some permutation of $(1, 2, 3, \ldots, k)$. We have proved in statement (ii) that the stationary probabilities $\pi_{(\sigma_l(1,2,3,\ldots,k),n_{k+1}-1,\ldots,n_N-1)}$ in (51) are all equal. Therefore, the last equation (52) is obtained and finally we prove that

$$\pi_{(1,2,3,\ldots,k,n_{k+1},\ldots,n_N)} = \pi_{(1,2,3,\ldots,k,n_{k+1}-1,\ldots,n_N-1)} \left(\sum_{i=1}^{k} c_i\right)$$

This completes the proof of the Lemma 2. □

So far, for any N file ages $(n_1, n_2, n_3, \ldots, n_N)$ we have derived the probability $\pi_{(n_1,n_2,n_3,\ldots,n_N)}$ in Theorem 4 when the updating system reaches the steady state. Taking the request popularity $\{p_i, 1 \leq i \leq N\}$ of cached files into consideration, the probability that the popularity-weighted average age Δ_{all} equals j can be determined as

$$\begin{aligned} &\Pr\{\Delta_{all} = j\} \\ &= \sum_{p_1 n_1 + p_2 n_2 + \cdots + p_N n_N = j} \pi_{(n_1,n_2,n_3,\ldots,n_N)} \\ &= \sum_{p_1 n_1 + p_2 n_2 + \cdots + p_N n_N = j} c_{g_\sigma^{-1}(N-1)} \prod_{j=1}^{N-1} c_{g_\sigma^{-1}(j-1)} \left(\sum_{k=1}^{j} c_{g_\sigma^{-1}(k-1)}\right)^{n_{g_\sigma^{-1}(j)} - n_{g_\sigma^{-1}(j-1)} - 1} \end{aligned} \tag{53}$$

Although a summation over certain state vectors is included in (53), we show that it is not hard to determine these states by numerical calculation. Thus, expression (53) almost gives the probability distribution function of Δ_{all}.

4.3. Obtaining the distribution function of the popularity-weighted average age for more general cached files updating models

In the previous part of this Section, we showed that by establishing an N-dimensional stochastic process, which describes the changes of N file ages simultaneously, the distribution function of the popularity-weighted average age over all the cached files can be determined. We obtain the mean $\overline{\Delta}$, and the distribution function for the general case where N files are generated at the server. More importantly, the idea that creating a discrete stochastic process can be applied for the updating-cached-files model with a more general setting. For example, assume that at each time slot multiple files can be refreshed at a time, or the server does not update any file. Allowing several

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 September 2020 doi:10.20944/preprints202009.0113.v1

updates at the same time will dramatically increase the number of possible state-vectors, but we show that the random transitions of N file ages can still be described by the stationary equations of the newly established age-process. In the following, we shall provide the stationary equations for this case without further calculation.

Now, let the cache size be N. We consider the cached-files-updating problem assuming that the server can refresh multiple files at each time slot. An N-dimensional random process is established for this case and only its stationary equations are given. Finding the closed-form solutions to the stationary probabilities are not included in this paper. We may solve the stationary equations in further work.

At each time slot, suppose that the server can update l files randomly, $0 \leq l \leq L$, making the ages of these files reset to 1. The refreshing probability distribution is defined as $\{c_I, I \in \mathcal{A}\}$, where $\mathcal{A}$ denotes the collection of all the subsets of $\mathcal{N} = \{f_1, f_2, \ldots, f_N\}$ whose size is no greater than L. For the sake of simplicity, the probability c_I can also be determined by the set of indices of file in set I. That is to say, for a set $I = \{f_{i_1}, f_{i_2}, \ldots, f_{i_m}\}$, $m \leq L$, we have $c_I = c_{\{i_1, i_2, \ldots, i_m\}}$. The probability $c_\emptyset$ corresponds to the case in which no files are updated, so that all the file ages become larger by one in this time slot. Define the N-dimensional random process

$$Age_N^{(L)} = \{(a_{1k}, a_{2k}, \ldots, a_{Nk}), a_{ik} \in \mathbb{N}, 1 \leq i \leq N, k \geq 1\}$$

where a_{ik} represents the age of the file f_i at the kth time slot.

At the current time slot, assume that the N file ages form the state $s = (n_1, n_2, \ldots, n_N)$. Remember that we assume that the files from the server are always new and transmission of each file from server to cache consumes exactly one time slot. This ensures that at the next time slot the age of any file f_i will jump to either $n_i + 1$ or 1, depending on whether this file is updated at current time slot. Define another state vector $s' = (n_1', n_2', \ldots, n_N')$ where $n_i' = n_i \mathbb{I}_{\{f_i \notin A\}} + 1, 1 \leq i \leq N$. We use A to denote the set of files that are refreshed by the server. The indicator function $\mathbb{I}(\cdot)$ is defined as

$$\mathbb{I}_{\{f_i \notin A\}} = \begin{cases} 1 & \text{if } f_i \notin A \\ 0 & \text{otherwise} \end{cases}$$

Thus, the single-step transition probability $P_{s,s'}$ that the state vector changes from s to s' is equal to

$$P_{s,s'} = c_A, \qquad A \in \mathcal{A}$$

Since the server can update several files in a time slot, at most L files in cache have age 1. In addition, it is also possible that all the file ages are greater than 1, because with a non-zero probability $c_\emptyset$ the server does not refresh any file. The characteristics of the age-vectors in the state space of process $Age_N^{(L)}$ can also be considered, like we do in Proposition before.

Now, in a state vector there are at most L components equal to 1. Without loss of generality, assume that all these "1" occur in the first L locations of the state vector s. The case that some "1" are contained in the last $(N - L)$ components can be converted to the former case by using certain vector permutations, and can be discussed accordingly. Suppose that the updating system reaches steady-state, let π_s be the stationary probability of the state vector s.

First of all, for the case $s = (n_1, n_2, \ldots, n_N)$ where $n_i \geq 2, 1 \leq i \leq N$, we show that

$$\pi_{(n_1, n_2, \ldots, n_N)} = \pi_{(n_1 - 1, n_2 - 1, \ldots, n_N - 1)} c_\emptyset \tag{54}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 September 2020 doi:10.20944/preprints202009.0113.v1

Next, let $s = (1, \ldots, 1, n_{k+1}, \ldots, n_N)$ in which the ages of first k, $1 \leq k \leq L$ files equal to 1 and assume that the latter $(N-k)$ ages are all greater than one. Then, for $1 \leq k \leq L$, the probability $\pi_{(1,\ldots,1,n_{k+1},\ldots,n_N)}$ can be determined as

$$\pi_{(1,\ldots,1,n_{k+1},\ldots,n_N)} = \left(\sum_{j_1,\ldots,j_k} \pi_{(j_1,\ldots,j_k,n_{k+1}-1,\ldots,n_N-1)}\right) c_{\{1,2,\ldots,k\}} \tag{55}$$

The sum on the right-hand side of (55) is taken over all the states whose last $(N-k)$ file ages are fixed to $(n_{k+1}-1, \ldots, n_N-1)$ and the first k components are arbitrary, as long as they can be k file ages. Considering all the permutations of state $(1, \ldots, 1, n_{k+1}, \ldots, n_N)$ and combining with (54), we give the stationary equations of the process $Age_N^{(L)}$ in Theorem 5 below.

Theorem 5. *Assume that at each time slot the server can refresh an arbitrary k, $0 \leq k \leq L$ files with probability c_A, where A denotes the set of k files updated by the server. Let the N-dimensional vector $s = (n_1, n_2, \ldots, n_N)$ be the state, which is composed of N file ages, we define an N-dimensional discrete random process $Age_N^{(L)} = \{s_t = (a_{1t}, a_{2t}, \ldots, a_{Nt}), t \geq 1\}$. The stationary equations for the process are written as*

$$\begin{cases} \pi_{(n_1,n_2,\ldots,n_N)} = \pi_{(n_1-1,n_2-1,\ldots,n_N-1)} c_{\emptyset} & (n_i \geq 2, 1 \leq i \leq N) \\ \pi_{\sigma_N(1,\ldots,1,n_{k+1},\ldots,n_N)} = \left(\sum_{j_1,\ldots,j_k} \pi_{\sigma_N(j_1,\ldots,j_k,n_{k+1}-1,\ldots,n_N-1)}\right) c_{k,\sigma_N} & (1 \leq k \leq L) \end{cases}$$

where σ_N denotes a permutation operator on the N-dimensional vector. For a given k, c_{k,σ_N} is the updating probability which is determined by the k files refreshed by the server. These k files can be found by virtue of σ_N.

5. Conclusion

In this paper, for the general cached-files-updating model, we determine the distribution function of the popularity-weighted average age over all the cached files. We show that an N dimensional stochastic process can be constituted to simultaneously describe the random transfers of N file ages. The stationary probability for an arbitrary state vector is found by solving the stationary equations of the resulting process. Thus, the distribution function of the popularity-weighted average age for arbitrarily given file popularities can be obtained by mergering a proper group of stationary probabilities. At the last of this paper, some more general models of cached files updating are also discussed. In particular, for the case where the server can update several files at one time slot, we give the stationary equations of the discrete process constituted for this case. For the further work, we intend to consider whether or not the distribution function of popularity-weighted average age can be obtained, assuming that the file popularities and the updating probabilities of the files are dependent and even time-varying.

Funding: Please add: "This research received no external funding" or "This research was funded by NAME OF FUNDER grant number XXX." and and "The APC was funded by XXX". Check carefully that the details given are accurate and use the standard spelling of funding agency names at https://search.crossref.org/funding, any errors may affect your future funding.

Acknowledgments: In this section you can acknowledge any support given which is not covered by the author contribution or funding sections. This may include administrative and technical support, or donations in kind (e.g., materials used for experiments).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Roy D. Yates, Y.S.; III, D.R.B.; Kaul, S.K.; Modiano, E.; Ulukus, S. Age of Information: An Introduction and Survey. *arXiv:2007.08564* **2020**.
2. Roy D. Yates, P.C.; Yener, A.; Wigger, M. Age-optimal constrained cache updating. *IEEE International Symposium on Information Theory* **2017**.

3. Jing Zhong, R.D.Y.; Soljanin, E. Two Freshness Metrics for Local Cache Refresh. *IEEE International Symposium on Information Theory* **2018**.
4. Haoyue Tang, P.C.; Wang, J.; Wigger, M.; Yates, R.D. Age of Information Aware Cache Updating with File- and Age-Dependent Update Durations. *arXiv:1909.05930v2* **2019**.
5. Ahani Ghafour, Y.D. Accounting for Information Freshness in Scheduling of Content Caching. *arXiv:1910.13194v1* **2019**.
6. Qing He, D.Y.; Ephremides, A. Optimizing freshness of information: On minimum age link scheduling in wireless systems. *International Symposium on Modeling and Optimization in Mobile, Ad-Hoc and Wireless Networks (WiOpt)* **2016**.
7. Qing He, D.Y.; Ephremides, A. On optimal link scheduling with min-max peak age of information in wireless systems. *IEEE International Conference on Communications(ICC)* **2016**.
8. Qing He, D.Y.; Ephremides, A. On optimal link scheduling with deadlines for emptying a wireless network. *IEEE International Symposium on Information Theory(ISIT)* **2017**.
9. Yu-Pin Hsu, E.M.; Duan, L. Age of Information: Design and Analysis of Optimal Scheduling Algorithms. *2017 IEEE International Symposium on Information Theory (ISIT)* **2017**.
10. Igor Kadota, E.U.B.; Singh, R.; Modiano, E. Minimizing the Age of Information in broadcast wireless networks. *54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)* **2017**.
11. Igor Kadota, A.S.; Uysal-Biyikoglu, E.; Singh, R.; Modiano, E. Scheduling Policies for Minimizing Age of Information in Broadcast Wireless Networks. *IEEE/ACM Transactions on Networking* **2018**, *26*, 2637–2650.
12. Rajat Talak, I.K.; Karaman, S.; Modiano, E. Scheduling Policies for Age Minimization in Wireless Networks with Unknown Channel State. *arXiv:1805.06752* **2018**.
13. Songtao Feng, J.Y. Age-Optimal Transmission of Rateless Codes in an Erasure Channel. 2019.
14. Ahmed Arafa, R.D.Y.; Poor, H.V. Timely Cloud Computing: Preemption and Waiting. *arXiv:1907.05408v1* **2019**.
15. Xi Zheng, S.Z.; Niu, Z. Context-Aware Information Lapse for Timely Status Updates in Remote Control Systems. *arXiv:1908.04446v1* **2019**.
16. Jingzhou Sun, Z.J.; Zhou, S.; Niu, Z. Optimizing Information Freshness in Broadcast Network with Unreliable Links and Random Arrivals: An Approximate Index Policy. *arXiv:1903.03723v1* **2019**.
17. Arunabh Srivastava, A.S.; Jagannathan, K. On Minimizing the Maximum Age-of-Information For Wireless Erasure Channels. *arXiv:1904.00647v1* **2019**.
18. Anis Elgabli, H.K.; Krouka, M.; Bennis, M. Reinforcement Learning Based Scheduling Algorithm for Optimizing Age of Information in Ultra Reliable Low Latency Networks. *arXiv:1811.06776v4* **2019**.
19. Elif Tugce Ceran, D.G.; Gyorgy, A. Reinforcement Learning to Minimize Age of Information with an Energy Harvesting Sensor with HARQ and Sensing Cost. *arXiv:1902.09467v1* **2019**.
20. Egemen Sert, C.S.; Baghaee, S.; Uysal-Biyikoglu, E. Optimizing age of information on real-life TCP/IP connections through reinforcement learning. *IEEE 26th Signal Processing and Communications Applications Conference (SIU)* **2018**.
21. Elif Tugce Ceran, D.G.; Gyorgy, A. A Reinforcement Learning Approach to Age of Information in Multi-User Networks. *arXiv:1806.00336v1* **2018**.
22. Elif Tugce Ceran, D.G.; Gyorgy, A. Average Age of Information with Hybrid ARQ under a Resource Constraint. *arXiv:1710.04971v3* **2017**.
23. Mohamed A. Abd-Elmagid, A.F.; Dhillon, H.S.; Saad, W. Deep Reinforcement Learning for Minimizing Age-of-Information in UAV-assisted Networks. *arXiv:1905.02993v1* **2019**.
24. Vishrant Tripathi, R.T.; Modiano, E. Age of Information for Discrete Time Queues. *arXiv:1905.02993v1* **2019**.
25. Antzela Kosta, N.P.; Ephremides, A.; Angelakis, V. Non-linear Age of Information in a Discrete Time Queue: Stationary Distribution and Average Performance Analysis. *arXiv:2002.08798v1* **2020**.
26. G. H. Hardy, J.E.L.; Polya, G. Inequalities. *Cambridge University Press* **1952**.