Available online at www.sciencedirect.com
SCIENCE DIRECT®
Performance Evaluation 63 (2006) 15–35
PERFORMANCE EVALUATION
An International Journal
www.elsevier.com/locate/peva

Parallel downloads for streaming applications—a resequencing analysis

Yoav Nebat[a,b], Moshe Sidi[a,*]

[a] *Electrical Engineering Department, Technion, Haifa 32000, Israel*
[b] *Department of Electrical and Computer Engineering, University of California, San Diego, La Jolla, CA 92093, USA*

Received 6 April 2004; received in revised form 5 November 2004; accepted 15 November 2004
Available online 1 January 2005

Abstract

Several recent studies have proposed methods to accelerate the receipt of a file by downloading its parts from different servers in parallel. The schemes suggested in most proposed parallel download approaches focus on reducing the total download duration. For streaming applications a more crucial performance issue is the regularity of flow of data to the application. Burstiness in data arrival implies longer playback delays, a higher probability of interruption at the application and larger required memory space for resequencing, which is undesirable and may become prohibitive for mobile devices with limited resources. This paper formulates models for an approach based on receiving only one copy of each of the data blocks in a file, while different data blocks may be obtained from different sources. This approach allows more robust download rates even when the conditions for each server/path used may change rapidly. In the parallel download scenario, out-of-order arrivals at the receiving side are unavoidable. We present methods to keep out-of-order low to ensure a more regulated flow of data to the application. A good indicator to the severeness of out-of-order arrival is the resequencing-buffer occupancy. The paper focuses on the analysis of the resequencing-buffer occupancy distribution and on the analysis of the methods used to reduce the occupancy of the buffer.

Keywords: Parallel download; Mirror server; Resequencing; Out-of-order arrival

* Corresponding author.
E-mail address: moshe@ee.technion.ac.il (M. Sidi).

0166-5316/$ – see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.peva.2004.11.006

1. Introduction

Text files, pictures, web pages and similar resources are often available at several sites in the network. A user that needs such a resource, say a file, would like to take advantage of the existence of copies of the same file in a manner that will result in better performance (e.g., smaller download time).

The common approach to obtain a resource in today's networks is to open a single TCP connection to a single source where the resource is available and receive all the required data via this single connection. While being very simple, this approach results in slow downloads, since only a single source is used, unless the client's link is utilized to its capacity by this source. With this approach, the fact that the same data is available at more than one location can be exploited by choosing a source that may give advantageous download times—a task that is often complicated. Studies [1–4] propose techniques for gathering information relevant to server selection.

A more recent approach is to open multiple TCP connections to the various locations where the same data is available, and download the data redundantly in parallel from all the sources. This may be done by applying some coding, e.g., "Tornado Codes" or "Erasure Codes" [5–9]. Transmitting a file repeatedly to multicast groups that may be joined by any client that may need the file [5,8,10] takes advantage of the redundancy in sources and seems to be very efficient in the multi-destination case (many-to-many). In the single destination case (many-to-one) it may result in some overhead in the total amount of transmitted data. This in turn may result in greater delay due to congestion and higher paid prices when charges are for the total amount of data received. Moreover, the flow of data to the application will be rather bursty, due to extensive resequencing that may be needed. In fact, for the codes suggested in [6–8] the decoding may begin only after the data is received completely from all sources, resulting in additional delay and highly bursty arrivals to the application.

Another approach is based on the fact that a user should strive to minimize the total download time, while keeping the overall data transmitted as low as possible. This can be achieved by downloading different parts of a file from different sources. Note that with this approach data segments are likely to arrive out-of-order, which may cause large resequencing-buffer demands, and re-ordering, resulting in an unregulated flow of data to the application at the destination [11]. An example for this approach is described in [12], where download time was measured for cases where different parts of a file were requested and downloaded in parallel from several servers. The results indicated that for large enough files, if the data chunk requests from each source are dynamically updated according to network conditions, the download duration becomes shorter compared to a single source case, even if the "best" single source was to be chosen.

In the context of streaming applications the latter approach has additional advantages. By dynamically deciding what data segment should be requested from each source, the playback delay can be significantly reduced compared to schemes that reconstruct the file only after the entire resource was received. Also, this alleviates the need to store all the data at the receiving end, which may become an issue when the resource is large and the available memory is limited (e.g., portable device). In this context the benefits of source diversity are in enabling streaming by providing a sufficient download rate with much lower reception rate variance compared to the single source case.

For this scheme to be robust and work effectively, the sizes of the data chunks requested from the different sources and the sequence and timing for issuing requests should be adapted to the ever changing network conditions. Otherwise, the required buffer space may keep increasing steadily and the initial playback delay will become insufficient, resulting in interruption at the application.

This paper focuses on the latter approach according to which different data chunks are requested from different servers, and downloaded in parallel. The basic idea is to obtain, before and during the reception process, statistics on the current possible reception rate and path delay from all the relevant sources. This can be achieved by using end-to-end measurements, as described in [12]. Based on this information, the receiver dynamically determines the specific data chunk and its corresponding magnitude to be requested from each source for timely reception of the data, as required by the application. By clever management of the requests lower reception rate variance is achieved, which in turn results in a more regulated flow of data to the application. This enables shorter playback delays and smaller resequencing-buffers, while incurring a minimal and balanced load on the network.

The goals of this paper are to formulate suitable models for the approach described above and analyze the performance of these models in terms of the resequencing-buffer requirements for different data request strategies. In our model we assume a fine, packet level granularity for requests made to the servers. Every packet in a downloaded file is assigned to (i.e., requested from) a *single* source responsible for its transmission. With this level of granularity any loss of generality in terms of possible data request strategies is avoided. In this paper, we derive the probability distribution of the resequencing-buffer occupancy for several possible download strategies. This is important for the determination of necessary buffer sizes at the receiver, and accordingly the required playback delay to keep potential interruption at the application under a specific threshold. We also devise algorithms for assigning which packets to download from each source, to help in keeping the resequencing-buffer occupancy as low as possible, and the data flow to the application as smooth as possible. Performance analysis of these algorithms is also presented.

The rest of this paper in organized as follows. In Section 2 we present the sources, destination and delay models. In particular, we derive a packet delay model for a single source that guarantees that packets it sends arrive in order to the destination. Section 3 contains the analysis for isochronous sources (sources that have equal transmission rates), while Section 4 contains the analysis for heterogeneous sources. Section 5 introduces algorithms for reducing the resequencing-buffer occupancy. Section 6 contains some numerical examples. Finally, Section 7 contains a summary and discussion.

2. The model

2.1. The sources and the destination

We consider the case where N sources transmit different segments of the same file to a destination. Let $\mathcal{L}$ denote the file to be downloaded and $\mathcal{L}_i$ $(1 \leq i \leq N)$ be the segment of the file to be downloaded from source i. To insure that each segment of the file is downloaded from a single source we need that $\mathcal{L}_i \cap \mathcal{L}_j = \emptyset$ for all $i \neq j$. Clearly, $\cup_i \mathcal{L}_i = \mathcal{L}$. An example for a file partition is for $\mathcal{L}_i$ to contain packets $i + lN$ for $l = 0, 1, 2, \ldots$.

One of the main problems dealing with the situation of receiving data segments of the same file from different sources simultaneously is that to forward the data to the application in sequence, the receiver must store packets received out of sequence in a *resequencing-buffer* or as separate data files in memory until all the preceding data is received. When data is received from several sources, packets may be received out of sequence, even if packets received from any single source are received in the order they were sent.

Due to the random delay, the occupancy of the resequencing-buffer at the destination will be random. Our goal is to determine the occupancy probabilities of this buffer. Having the probability distribution of the buffer occupancy will enable the provision of necessary buffer size at the receiver to keep potential overflows under a specified threshold.

2.2. The delay from source to destination

Consider the random delay packets experience when transmitted from a source to the destination. We focus on delay models that ensure packets from a single source arrive in the order transmitted despite the random delay. This is often the case when a single static route from the source to the destination is used. Other delay models, including an analysis of the resequencing-buffer occupancy while receiving data from a single source (via possibly lossy routes), are studied in [13].

In the following we derive the basic necessary conditions upon the delay distribution to ensure packets arrive at their destination in the same order they were transmitted. To facilitate the explanation, we use the following notations:

$T(i)$—the time packet i was transmitted;
$d(i)$—the delay experienced by packet i;
$A(i)$—the time packet i arrived at the destination;
$\Delta_{\mathrm{T}}(i)$—the inter-departure time for packets $i-1$ and i;
$\Delta_{\mathrm{A}}(i)$—the inter-arrival time for packets $i-1$ and i.

Obviously, $A(i) = T(i) + d(i)$, $\Delta_{\mathrm{T}}(i) = T(i) - T(i-1)$ and $\Delta_{\mathrm{A}}(i) = A(i) - A(i-1)$. To ensure arrival in sequence it is necessary and sufficient to ensure that for all $i \geq 0$ packet $i+1$ will not arrive before packet i. The time packet i is received can be written as $A(i) = T(i-1) + \Delta_{\mathrm{T}}(i) + d(i)$, while the time packet $i-1$ is received is $A(i-1) = T(i-1) + d(i-1)$. For orderly reception we require that $\Delta_{\mathrm{A}}(i) = A(i) - A(i-1) \geq 0\ \forall i$. This condition can be written as $\Delta_{\mathrm{A}}(i) = A(i) - A(i-1) = T(i-1) + \Delta_{\mathrm{T}}(i) + d(i) - T(i-1) - d(i-1) = \Delta_{\mathrm{T}}(i) + d(i) - d(i-1) \geq 0\ \forall i$, or $d(i) \geq d(i-1) - \Delta_{\mathrm{T}}(i)$. We also need $d(i)$ to be non-negative, $d(i) \geq 0\ \forall i$.

Consequently, the only restrictions upon $d(i)$ are: $d(i) \geq 0$ and $d(i) \geq d(i-1) - \Delta_{\mathrm{T}}(i)$. Therefore, $d(i)$ can be any probabilistic function $f(d(i-1), \Delta_{\mathrm{T}}(i))$ that satisfies

$$f(d(i-1), \Delta_{\mathrm{T}}(i)) \geq \begin{cases} d(i-1) - \Delta_{\mathrm{T}}(i) & d(i-1) - \Delta_{\mathrm{T}}(i) \geq 0 \\ 0 & d(i-1) - \Delta_{\mathrm{T}}(i) < 0 \end{cases} \tag{1}$$

To ensure the delay does not diverge, we need a stability constraint on $f(d(i-1), \Delta_{\mathrm{T}}(i))$.

For the special case where packets are transmitted at a constant rate (as is suitable for streaming), i.e., every time unit, we have $\Delta_{\mathrm{T}}(i) = 1\ \forall i \geq 1$, and (1) becomes

$$f(d(i-1), \Delta_{\mathrm{T}}(i)) = f(d(i-1)) \geq \begin{cases} d(i-1) - 1 & d(i-1) \geq 1 \\ 0 & d(i-1) < 1 \end{cases}$$

For simplicity, let the delay be integer multiples of a time unit. Consequently, $f(d(i-1))$ can be any integer that satisfies

$$d(i) = f(d(i-1)) \geq \begin{cases} d(i-1)-1 & d(i-1) > 0 \\ 0 & d(i-1) = 0 \end{cases} \tag{2}$$

Consider the Markov-chain in Fig. 1 where state i corresponds to delay of i time units and the arrows correspond to the transitions among states with the respective probabilities (i.e. p_{ij} is the probability of having delay j time units for the next packet, given that the current packet is delayed i time units). Any delay process represented by this Markov-chain assures (2) is satisfied. Also, assuming disjoint paths from sources to destination, this model is quite general and reasonable to assume.

For stability of the delay we need (see Theorem 4 in [14]) that there exists M, such that

$$p_{i(i-1)} \geq \sum_{k=i+1}^{\infty} (k-i)p_{ik}, \quad \forall i \geq M \tag{3}$$

which is sufficient to ensure the irreducible and aperiodic Markov-chain that represents the delay process is recurrent.

In [15] Baccelli and Makowski state the result that if the arrival process into a disordering network (equivalent to the transmission instances in our model), and the delay in the disordering network are jointly ergodic, the resequencing queue is assured to be stable (i.e., steady-state occupancy probabilities for the resequencing-buffer exist), if orderly arrivals are expected.

In the multi-source case, several Markov delay models similar to the above are combined. This can be easily shown to establish a multi-dimensional recurrent Markov process for the set of delays of packets from the different sources. Hence, the delay of packets in the disordering network is ergodic. Combined with the constant transmission rates of the sources, ergodicity is assured.

However, whether the arrival into the disordering network is orderly depends on the assignment of packets to be transmitted by the different sources. The cyclic assignment in Section 3 assures global orderly transmission. In Section 4, an assignment based on *Sturmian Words*, assures global orderly transmission. In Section 5, the suggested algorithms result in disorderly global transmission. However, the analysis shows that these assignments have equivalent cyclical assignments for which all the sufficient conditions above hold. Hence, throughout this paper, all resequencing queues are stable and can therefore be analyzed in terms of steady state occupancy probabilities.

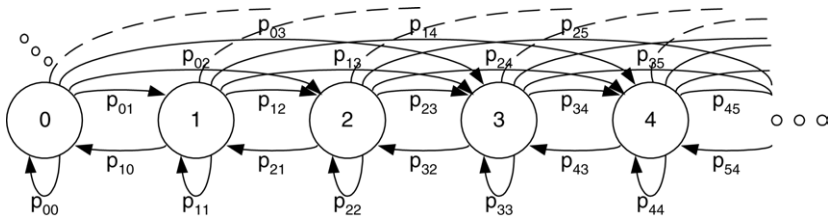

Fig. 1. The Markov-chain representing the delay.

3. Isochronous sources

In this section we consider isochronous sources, i.e., sources with transmission rates equal to one packet per time unit. We begin by considering a simple partition of a file in which the packets transmitted at time 0 by sources $1, 2, \ldots, N$ are packets $1, 2, \ldots, N$, respectively. After transmitting the first N packets, the packets are appointed to the sources cyclically, thus, at time t sources $1, 2, \ldots, N$ transmit packets $1 + Nt, 2 + Nt, \ldots, N + Nt$, respectively, i.e., $\mathcal{L}_i$ contains packets $i + lN$ for $l = 0, 1, 2, \ldots$.

The delay of packets from each source is a random process that corresponds to the general discrete delay model for packets received in sequence from a single source. The delay processes for the different sources are represented by (possibly different) realizations of the Markov-chain in Fig. 1. This allows for unique average delay and the steady-state probabilities for each source. Assuming disjoint paths are used for transmission, the delays from different sources are independent. Note that we allow zero delays in our model.

3.1. Two sources

To facilitate the presentation we begin with a two-source system. In this case source 1 transmits the odd indexed packets (i.e., 1, 3, 5, ...) while source 2 transmits the even indexed packets (i.e., 2, 4, 6, ...). To find the distribution of the resequencing-buffer occupancy we assume steady-state is maintained before time t, and calculate the resequencing-buffer occupancy probabilities at time t. The following definitions are required:

Definition. $d_{\mathrm{X},t}$ is the delay experienced by the last packet received from source X at time t.

Definition. $\Delta_{\mathrm{X},t}$ is the time that passed since receipt of the last packet from source X at time t.

Definition. $\delta_{\mathrm{X},t}$ is the time that passed since the last packet received from source X was transmitted at time t.

For brevity, in the remaining of the paper we refer to $d_{\mathrm{X},t}$, $\Delta_{\mathrm{X},t}$ and $\delta_{\mathrm{X},t}$ as d_{X}, Δ_{X} and δ_{X}, respectively, keeping in mind the values are all time dependent. The value d_{X} may change upon receipt of a packet from source X, and its value is set to the value of the delay of the packet that caused the change. If more than one packet transmitted by source X arrive simultaneously, d_{X} obtains the value of the delay of the packet most recently sent. The value Δ_{X} is updated every time unit. If a packet from source X has arrived $\Delta_{\mathrm{X}} = 0$, otherwise Δ_{X} is increased by 1.

Fig. 2 depicts a typical scenario for two sources. We can see that at $t = 8$ the last packet received from source 1 is packet 7. This packet was transmitted at $t = 3$ and arrived at the destination at $t = 5$, therefore, at $t = 8$ we have $d_1 = 2$. No packet from source 1 was received between $t = 5$ and $t = 8$, therefore, at $t = 8$, $\Delta_1 = 3$. On the other hand, the last packets received from source 2 at $t = 8$ are packets 8 and 10. The most recently transmitted packet among these packets (packet 10) was transmitted at $t = 4$ and arrived at $t = 8$, therefore, at $t = 8$ we have: $d_2 = 4$ and $\Delta_2 = 0$. From Fig. 2 and the definition of δ_{X} it is easy to conclude that $\delta_{\mathrm{X}} = d_{\mathrm{X}} + \Delta_{\mathrm{X}}$.

Theorem 1. *Given δ_1 and δ_2 at time t, the resequencing-buffer occupancy at time t, B, is given by*:

$$B = \begin{cases} \delta_1 - \delta_2 & \delta_1 \geq \delta_2 \\ \delta_2 - \delta_1 - 1 & \delta_1 < \delta_2 \end{cases} \tag{4}$$

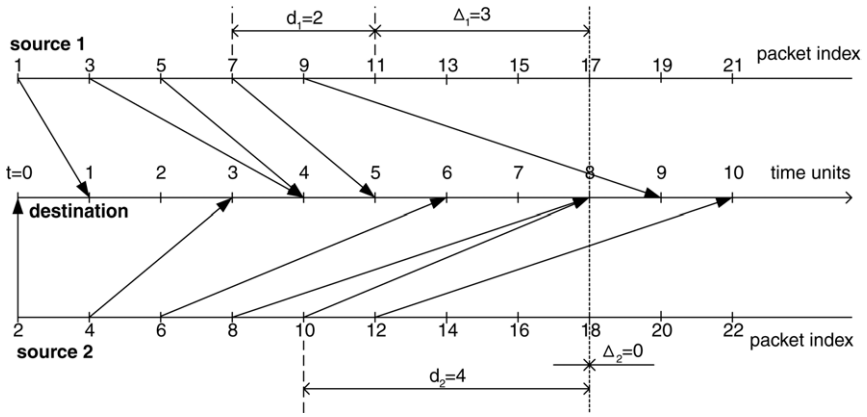

Fig. 2. d_1, d_2, Δ_1 and Δ_2 in a typical scenario.

Proof. At time t source 1 transmits packet $1+2t$, and source 2 transmits packet $2+2t$. Given δ_1 and δ_2, the last packet received from source 1 is packet $L_1 = 1+2t-2\delta_1$, and the last packet received from source 2 is packet $L_2 = 2+2t-2\delta_2$.

If $\delta_1 \geq \delta_2$ then $L_2 > L_1$. Therefore, if there are any packets in the resequencing-buffer they are even indexed packets transmitted by source 2, and the resequencing-buffer occupancy is

$$B = \frac{L_2 - (L_1+1)}{2} = \delta_1 - \delta_2 \tag{5}$$

since packet L_1+1 (transmitted by source 2) is not stored in the resequencing-buffer, because all its preceding packets have arrived. All the packets that were transmitted by source 2 after packet L_1+1, and arrived till time t are stored in the resequencing-buffer. Since no other packets transmitted by source 2 and no packet transmitted by source 1 are stored in the resequencing-buffer, (5) holds.

If $\delta_1 < \delta_2$ then $L_1 > L_2$, therefore, if there are any packets in the resequencing-buffer they are odd indexed packets transmitted by source 1, and the resequencing-buffer occupancy is

$$B = \frac{L_1 - (L_2+1)}{2} = \delta_2 - \delta_1 - 1 \tag{6}$$

since packet L_2+1 (transmitted by source 1) is not stored in the resequencing-buffer, because all its preceding packets have arrived. All the packets that were transmitted by source 1 after packet L_2+1, and arrived till time t are stored in the resequencing-buffer. Since no other packets transmitted by source 1 and no packet transmitted by source 2 are stored in the resequencing-buffer, (6) holds. Therefore, (4) holds, and the proof is completed. □

From (4) we conclude that the steady-state probability of having k packets ($k = 0, 1, 2, \ldots$) stored in the resequencing-buffer is

$$P(B=k) = P(\delta_1 - \delta_2 = k) + P(\delta_2 - \delta_1 - 1 = k) \tag{7}$$

Since δ_1 depends solely on the delay of packets transmitted by source 1, and, δ_2 depends solely on the delay of packets transmitted by source 2, δ_1 and δ_2 are independent. Using $\delta_1 \geq 0$ and $\delta_2 \geq 0$, (7) becomes

$$P(B=k)=\sum_{x=0}^{\infty} P(\delta_2=x)P(\delta_1=x+k)+\sum_{x=0}^{\infty} P(\delta_1=x)P(\delta_2=x+1+k) \tag{8}$$

To find the steady-state probabilities $P(\delta_1 = x)$ and $P(\delta_2 = x)$, $x = 0, 1, 2, \ldots$ we shall prove the following important property of δ_X.

Theorem 2. *The steady-state probability distribution of δ_X and the probability distribution of the delay of packets transmitted by source X, d_X, are identical.*

Proof. In order to prove the above theorem we shall prove that $P(\delta_X = k) = P(d_X = k)$, $k = 0, 1, 2, \ldots$. The theorem considers a single source, source X. For brevity, the subscripts X are omitted in the proof.

The event $\delta = k$ is equivalent to the event $A(n) \leq t < A(n+1)$, where n is uniquely determined by requiring that $t = T(n) + k$. Therefore

$$\begin{aligned} P(\delta=k) &= P(A(n)\leq t<A(n+1)) = P(A(n)\leq T(n)+k<A(n+1)) \\ &= P(d(n)\leq k, d(n+1)>k-1) = P(d(n)\leq k\leq d(n+1)) = P(d(n)=k) \end{aligned} \tag{9}$$

where the last equality follows from the balance equations of the Markov-chain in 1.

Therefore, the equality

$$P(\delta_X = k) = P(d_X = k),\ k \geq 0 \tag{10}$$

holds, and the proof of Theorem 2 is completed. □

Let $P_X(i)$ be the steady-state probability of the delay of a packet transmitted by sources X being i, $i \geq 0$. Substituting (10) into (8) we have

$$P(B=k)=\sum_{i=0}^{\infty} P_2(i)P_1(i+k)+\sum_{i=0}^{\infty} P_1(i)P_2(i+k+1) \tag{11}$$

For the simple example where the delay of packets transmitted by sources 1 and 2 are identical, and can be represented by the birth–death Markov-chain in Fig. 3, the steady-state probabilities of the delay are

$$P_1(i)=P_2(i)=\begin{cases}(1-\alpha)(1-\rho)\rho^{i-1} & i\geq 1\\ \alpha & i=0\end{cases}$$

where $\alpha \triangleq \dfrac{p_- - p_+}{p_- - p_+ + p_1}$ and $\rho \triangleq \dfrac{p_+}{p_-}$ where, according to (3), $\rho < 1$ assures stability for the delay model.

Substituting $P_1(i)$ and $P_2(i)$, $i \geq 0$ in (11) we obtain

$$P(B=k)=\begin{cases}(1-\alpha)(1-\rho)(\rho+\alpha)\rho^{k-1} & k\geq 1\\ \alpha^2+(1-\alpha)(1-\rho) & k=0\end{cases}$$

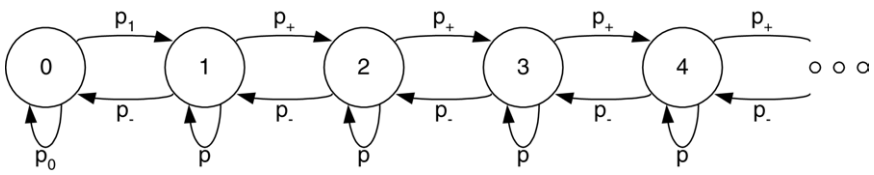

Fig. 3. The simple Markov-chain representing the delay.

3.2. N sources

We consider the case where N sources transmit different chunks of the same file to a destination. Sources transmit at an equal rate of one packet per time unit. The packets transmitted at time 0 by sources $1, 2, \ldots, N$ are packets $1, 2, \ldots, N$ respectively. After transmitting the first N packets, the packets are appointed to the sources cyclically, thus, at time t sources $1, 2, \ldots, N$ transmit packets $1 + Nt, 2 + Nt, \ldots, N + Nt$ respectively.

The delay processes for the different sources are represented by different realizations of the Markov-chain in Fig. 1.

The derivation of the resequencing-buffer occupancy for N sources is similar to the two-source case. Yet, the notion of the minimum valued packet (mvp) is helpful here. The mvp at time t is the lowest indexed packet that has not arrived at the destination's receiver by time t (see [16]). For instance, if packets 1 through 15 arrived, but packet 16 did not, packet 16 is the mvp.

From this definition it is easy to observe that: (i) the resequencing-buffer occupancy at time t is exactly the number of packets indexed higher than the mvp, that have arrived by time t; (ii) no packet stored in the resequencing-buffer was transmitted by the source of the mvp, since packets transmitted by every source arrive in transmission order.

Denote the index of the source of the mvp by s_m.

Lemma 3. *$s_m = i$ if and only if $\delta_i \geq \delta_j \forall j \in \{1, 2, \ldots, N\}$ and $i < j \forall j \mid \delta_j = \delta_i$, $j \neq i$.*

Proof. It is easily verified that $s_m = i \Rightarrow \delta_i \geq \delta_j \forall j \in \{1, 2, \ldots, N\}$ and $i < j \forall j \mid \delta_j = \delta_i$, $j \neq i$.

To understand why $\delta_i \geq \delta_j \forall j \in \{1, 2, \ldots, N\}$ and $i < j \, \forall j \mid \delta_j = \delta_i$, $j \neq i \Rightarrow s_m = i$, assume source i satisfies the above inequalities, but $s_m = k$, $k \neq i$.

If we denote the lowest indexed packet transmitted by source X that did not arrive at the destination's receiver by time t by $m^x(t)$, for sources i and k we have: $m^i(t) = i + N(t - \delta_i + 1)$ and $m^k(t) = k + N(t - \delta_k + 1)$, respectively, as shown in Fig. 4.

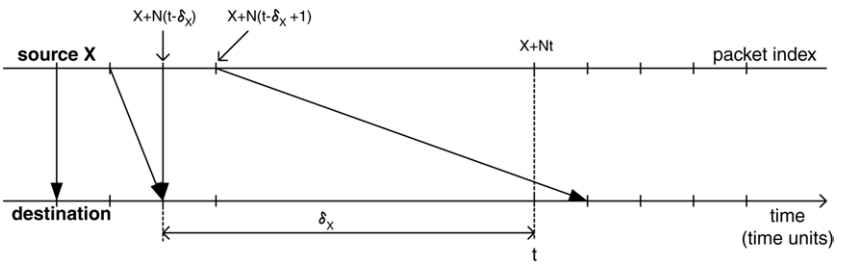

Fig. 4. δ_X and the corresponding lowest indexed packet transmitted by source X that did not arrive at the destinations' receiver.

If $k > i$, using the fact that $\delta_i \geq \delta_k$ we get $m^k(t) = k + N(t - \delta_k + 1) \geq k + N(t - \delta_i + 1) > i + N(t - \delta_i + 1) = m^i(t)$, therefore $m^i(t) < m^k(t)$, contradicting the assumption that $s_m = k$.

On the other hand, if $k < i$, using the fact that $\delta_i > \delta_k$, we get $m^k(t) = k + N(t - \delta_k + 1) = (k + N) + N(t - (\delta_k + 1) + 1) \geq (k + N) + N(t - \delta_i + 1) > i + N(t - \delta_i + 1) = m^i(t)$, therefore, $m^i(t) < m^k(t)$, again, contradicting $s_m = k$. Therefore, we conclude that $s_m = i$. □

Lemma 4. *If $s_m = i$, the number of packets that were transmitted by source j and are stored in the resequencing-buffer, a_j, is*

$$a_j = \begin{cases} \delta_i - \delta_j & j > i \\ \delta_i - \delta_j - 1 & j < i \\ 0 & j = i \end{cases} \tag{12}$$

Proof. The index of the mvp is $m^i(t) = i + N(t - \delta_i + 1)$. The index of the last packet received from source j is $j + N(t - \delta_j)$.

If $j > i$, the lowest indexed packet transmitted by source j and indexed higher than the mvp is packet $j + N(t - \delta_i + 1)$, therefore, a_j is given by

$$a_j = \frac{j + N(t - \delta_j) - [j + N(t - \delta_i + 1)]}{N} + 1 = \delta_i - \delta_j - 1 + 1 = \delta_i - \delta_j \tag{13}$$

If $j < i$, the lowest indexed packet transmitted by source j and indexed higher than the mvp is packet $j + N(t - \delta_i + 2)$, therefore, a_j is given by

$$a_j = \frac{j + N(t - \delta_j) - [j + N(t - \delta_i + 2)]}{N} + 1 = \delta_i - \delta_j - 2 + 1 = \delta_i - \delta_j - 1 \tag{14}$$

As mentioned before, no packet stored in the resequencing-buffer was transmitted by the source of the mvp, therefore, for $j = i$ we have $a_j = 0$. Consequentially, (12) holds, and the proof is completed. □

Given that $s_m = i$, the resequencing-buffer occupancy is

$$B = \sum_{j=1}^{N} a_j = \sum_{j=1}^{i-1} (\delta_i - \delta_j - 1) + \sum_{j=i+1}^{N} (\delta_i - \delta_j) = (N-1)\delta_i + 1 - i - \sum_{j=1, j\neq i}^{N} \delta_j \tag{15}$$

The probability of having k packets in the resequencing-buffer is given by

$$P(B = k) = \sum_{i=1}^{N} \sum_{x=0}^{\infty} P(B = k, s_m = i, \delta_i = x). \tag{16}$$

The probability on the right hand side of (16), using (10) and (15) is

$$P(B = k, s_m = i, \delta_i = x) = P\left(\sum_{j=1, j\neq i}^{N} \delta_j = (N-1)x + 1 - i - k, \delta_j < x \forall j < i, \delta_j \leq x \forall j > i \right)$$

$$= \sum_{S_{i,x,k}} P_i(x) \prod_{j=1}^{i-1} P_j(l_j) \prod_{j=i+1}^{N} P_j(l_j) \tag{17}$$

where

$$S_{i,x,k} = \left\{ l_1, \ldots, l_{i-1}, l_{i+1}, \ldots, l_N : l_1 < x, \ldots, l_{i-1} < x, l_{i+1} \leq x, \ldots, l_N \leq x, \right.$$
$$\left. \sum_{j=1, j\neq i}^{N} l_j = (N-1)x + 1 - i - k \right\}.$$

4. Heterogeneous sources

In this section we focus on a system that consists of two sources with different transmission rates. Source 1 transmits a packets per time unit, while source 2 transmits b packets per time unit. In this case, assigning packets to be transmitted to the sources in the same manner as in Section 3.1 (i.e., odd indexed packets to source 1, and even indexed packets to source 2), will result in a constant growth of the difference in the indexes of the packets transmitted by the sources, which will cause a steadily growing resequencing-buffer occupancy (In terms of [15] this is a non-ergodic inter-arrival sequence, resulting in an unstable resequencing queue). Therefore, we should first devise a packet assignment that will result in steady buffer occupancy probabilities, and then find the occupancy probabilities.

With no loss of generality, assume a and b are integers, with no common divisor greater than 1. The delay of packets from each source is a random process that corresponds to the Markov-chain in Fig. 1.

4.1. Packet assignment

The assignment of packets to sources follows the rule that each packet is assigned to the source that would be able to transmit it the earliest. In the case both sources could transmit it at the same time, the packet is assigned to source 1, and the next packet in sequence is assigned to source 2.

Denote the index of the source packet n is assigned to by β_n. Define $\kappa \stackrel{\triangle}{=} \frac{b}{a+b}$ and $\eta \stackrel{\triangle}{=} b\left(1 - \frac{2}{a+b}\right)$. The assignment is determined by, e.g., the *Sturmian Word* (see [17]) $s_{\kappa,\eta} = \beta_1, \beta_2, \ldots$ where

$$\beta_n = \begin{cases} 1 \text{ if } \lceil \kappa(n+1) + \eta \rceil = \lceil \kappa n + \eta \rceil \\ 2 \text{ otherwise} \end{cases}$$

An example of the assignment obtained for $a = 5$ and $b = 3$ is presented in Fig. 5.

4.2. The resequencing-buffer occupancy

In the Markov-chains representing the delay of packets transmitted by source 1 and source 2, transitions occur upon every packet transmission. Therefore, the transition rates differ, corresponding to the transition rates of the sources. We define a source X time unit as the time between two subsequent packet transmissions of source X. Clearly, a (global) time unit is equal to a source 1 time units, or to b source 2 time units.

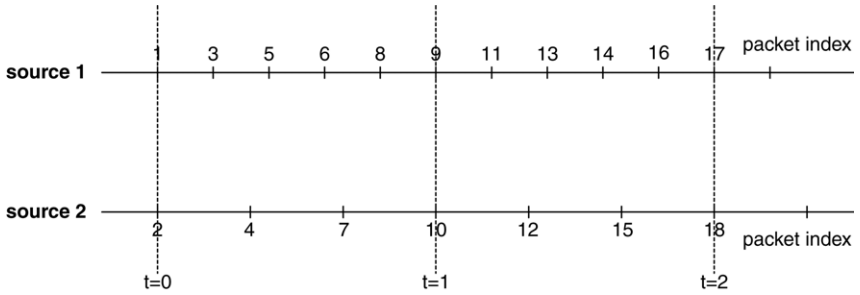

Fig. 5. Packet assignment for $a = 5$ and $b = 3$.

In this section we use the following definition for δ_X:

Definition. δ_X is the number of source X time units that passed since the last packet received from source X was transmitted.

Theorem 5. *The resequencing-buffer occupancy, at the instances that both sources transmit packets simultaneously* (*every* (*global*) *time unit*) *is*

$$B = \begin{cases} \left\lceil \dfrac{(\delta_2 - 1)a}{b} \right\rceil - \delta_1 & a\delta_2 - b\delta_1 > a - b \\ \left\lfloor \dfrac{(\delta_1 - 1)b}{a} + 1 \right\rfloor - \delta_2 & a\delta_2 - b\delta_1 \le a - b \end{cases} \tag{18}$$

(18) *is simply a generalization of* (4) *for the case of heterogeneous sources. For a rigorous proof refer to* Appendix A.

The probability of having k packets stored in the resequencing-buffer is

$$\begin{aligned} P(B = k) = &\sum_{v=0}^{\infty} P\left(\frac{(k + v - 1)b}{a} + 1 < \delta_2 \le \frac{(k + v)b}{a} + 1 \right) P(\delta_1 = v) \\ &+ \sum_{u=0}^{\infty} P\left(\frac{(k + u - 1)a}{b} + 1 \le \delta_1 < \frac{(k + u)a}{b} + 1 \right) P(\delta_2 = u) \end{aligned} \tag{19}$$

5. Reducing the buffer occupancy

Using the assignment method, as in Section 3, according to which packets $1, 2, \ldots, N$ are transmitted by sources $1, 2, \ldots, N$, respectively, and afterwards assigning packets cyclically, ignores any knowledge regarding the delay distributions.

To maintain a low resequencing-buffer occupancy, we present a new assignment method, taking delay distributions into consideration. In this section we assume an end-to-end measurement had taken place, and an estimate of the average delay from each source is available.

5.1. Packet assignment for two isochronous sources

We begin with two isochronous sources each transmitting with a rate of one packet per time unit. Let the delay of packets transmitted by each source be constant, i.e., packets transmitted by source 1 are delayed d_1 time units on their route to the destination, and packets transmitted by source 2 are delayed d_2 time units. Under a constant delay assumption, a packet from each source is received every time unit, thus, $\Delta_1 = \Delta_2 = 0$. As a result, $\delta_1 = d_1$ and, $\delta_2 = d_2$. Consequently, the resequencing-buffer occupancy, using the assignment method represented in Section 3.1 is, according to (4), given by

$$B = \begin{cases} d_1 - d_2 & d_1 \geq d_2 \\ d_2 - d_1 - 1 & d_1 < d_2 \end{cases} \tag{20}$$

From (20) we conclude that using the packet assignment as in Section 3.1 results in a constant buffer occupancy, that corresponds approximately to the delay difference, even though the delays are constant.

In the situation above, it would be much more advantageous to assign higher indexed packets to the source with the larger delay. For instance, assuming $d_1 > d_2$, and $d_1 - d_2 = \alpha$, if we assign packets $\alpha + 1, \alpha + 3, \alpha + 5, \ldots$ to source 1, and packets $1, 2, \ldots, \alpha, \alpha + 2, \alpha + 4, \ldots$ to source 2, the resequencing-buffer occupancy would be 0 throughout the receipt of the file, since packets 1 through α will be received in order, and then packets $\alpha + 1$ and $\alpha + 2$ will be received simultaneously, as well as packets $\alpha + 3$ and $\alpha + 4$, packets $\alpha + 5$ and $\alpha + 6$, etc.

From the example above we conclude that given the delay distributions we can achieve lower buffer occupancies if we take the delay distributions into consideration when we assign packets to sources.

The criterion for assignment in our algorithm is that packets are assigned to the sources in a manner that ensures that if all the packets experience a delay that corresponds to the average delay of packets transmitted by the source they are assigned to, the buffer occupancy would be 0 throughout the receipt of the file.

In the general case, when packet delay is random, let the average delays of packets transmitted by sources 1 and 2 be $\bar{d}_1$ and $\bar{d}_2$, respectively. We assume, without loss of generality, that $\bar{d}_1 \geq \bar{d}_2$, and define $\alpha = \lceil \bar{d}_1 - \bar{d}_2 \rceil$. Similar to the method shown for constant delays, we assign packets $\alpha + 1, \alpha + 3, \alpha + 5, \ldots$ to source 1, and, packets $1, 2, \ldots, \alpha, \alpha + 2, \alpha + 4, \ldots$ to source 2.

If we use this packet assignment and assume the delay for packets transmitted by each source is exactly the average delay from the transmitting source, all packets arrive in sequence. If packets $\alpha + 1$ and 1 are transmitted at time 0 by sources 1 and 2, respectively, and every source transmits a new packet assigned to it every time unit, packets $1, 2, \ldots, \alpha$ would arrive at time $\bar{d}_2, \bar{d}_2 + 1, \ldots, \bar{d}_2 + \alpha - 1$, respectively. Packet $\alpha + 1$ would arrive at time $\bar{d}_1, \bar{d}_2 + \alpha - 1 < \bar{d}_1 \leq \bar{d}_2 + \alpha$, and packets $\alpha + 3, \alpha + 5, \ldots$ would arrive at time $\bar{d}_1 + 1, \bar{d}_1 + 2, \ldots$, respectively. Packet $\alpha + 2$ would arrive at time $\bar{d}_2 + \alpha$, and packets $\alpha + 4, \alpha + 6, \ldots$ would arrive at time $\bar{d}_2 + \alpha + 1, \bar{d}_2 + \alpha + 2, \ldots$, respectively. Therefore, packet $\alpha + 1$ will be received before packet $\alpha + 2$, but after packet α, since $\bar{d}_2 + \alpha - 1 < \bar{d}_1 \leq \bar{d}_2 + \alpha$. Similarly, packet $\alpha + 3$ will be received before packet $\alpha + 4$, but after packet $\alpha + 2$, etc. Consequently, the resequencing-buffer occupancy would be 0 throughout the receipt of the file, since all packets would be received in order, in accordance to the assignment criterion.

Fig. 6 illustrates how the assignment provides an occupancy of 0, throughout the receipt of all the data, if all packets are delayed according to the average delay of packets transmitted by their source. In this example $\bar{d}_1 = 4.4$ and $\bar{d}_2 = 2.6$, therefore $\alpha = \lceil \bar{d}_1 - \bar{d}_2 \rceil = 2$. The packets assigned to source 1 are packets 3, 5, 7, ..., and the packets assigned to source 2 are 1 and 2, and then 4, 6, 8, ... From the figure

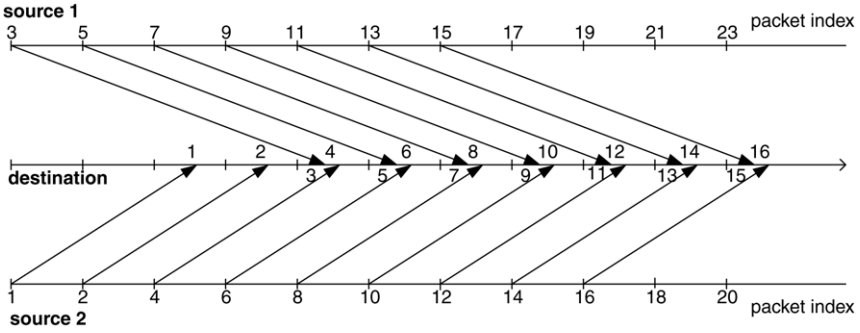

Fig. 6. Packet assignment to sources and arrival times.

we can see that packets arrive at the destination in sequence, therefore the resequencing-buffer occupancy is 0 throughout the receipt of the data.

Lemma 6. *For the proposed assignment algorithm, the resequencing-buffer occupancy probabilities are equal to the resequencing-buffer occupancy probabilities when packets are assigned cyclically, and the delay in all states of the Markov-chain representing the delay of packets transmitted by source* 2 *is increased by* α.

Proof. By increasing the delay in all states of the Markov-chain for source 2, the steady state probability of having a delay of $d + \alpha$ is equal to the steady state probability of having delay d in the original system. This shift of α time units in the delay compensates for the initial, non-cyclic assignment of packets in the proposed assignment algorithm, and makes the cyclic assignment equivalent to the new suggested assignment algorithm. For a more rigorous proof see [18]. □

Note that by shifting the delay and then assigning cyclically, the delays from the two sources become within one time unit from each other, on the average. The suggested assignment algorithm will result in performance that is identical to a system where the average delays are approximately equal and cyclic assignment is used.

Using Lemma 6 we can apply (11) in order to find the resequencing-buffer occupancy probabilities, when packets are assigned to sources according to the proposed assignment algorithm.

It is interesting to note that under the average buffer occupancy criterion, the proposed assignment algorithm does not assure improvement, i.e., although in general the algorithm helps reduce the average buffer occupancy, some rare pathological cases exist whereby the average occupancy is actually slightly increased by using the proposed assignment algorithm. Such increases, when they occur, tend to be negligible, therefore the possible benefits by far surpass the unlikely drawbacks.

5.2. Packet assignment for N isochronous sources

The algorithm suggested for the two source case uses a transient assignment that compensates for the different average delays, before it switches to the cyclic part, which determines the steady-state buffer occupancy probabilities. A similar algorithm for N sources, where after an initial transient, assignment proceeds in a periodic fashion that determines the steady-state occupancy of the buffer is suggested in [18].

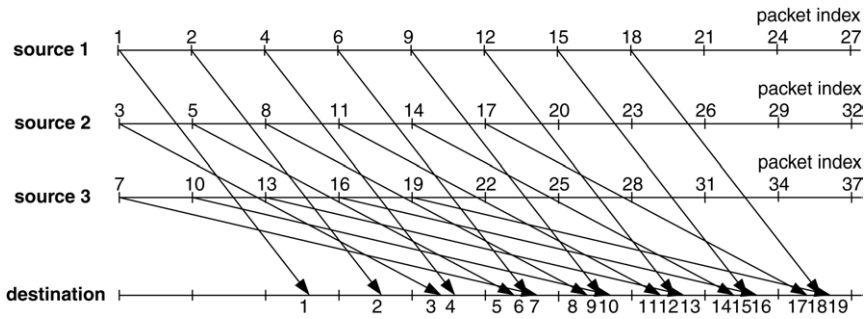

Fig. 7. Packet assignment to three sources and arrival times.

Fig. 7 illustrates how the assignment provides an occupancy of 0 throughout the receipt of all the packets, for a case where $N = 3$, if all packets are delayed according to the average delay of packets transmitted by their source. In this example the average delays from the sources are 2.6, 4.4 and 5.7.

Similar to the two source case, for the N source case, (16) can be utilized to compute the steady-state occupancy probabilities (see [18]).

6. Numerical results

Figs. 9–11 illustrate the buffer occupancy probabilities for several scenarios. In all cases, the delay process of each source is represented by a Markov-chain as in Fig. 8, with the transition probabilities and M as parameters of the source.

Fig. 9 illustrates the buffer occupancy probabilities for several cases of two-source scenarios. The solid line represents the case where the delay models used for both sources are uniform (this is achieved by using $p_+ = p_-$) in the range [0, 100] time units.

For the cases where for source 2, $p_+ = 0.35$, $p_- = 0.25$, we notice how using the proposed assignment algorithm helps in keeping the buffer occupancy low. The average occupancy drops from 46.3 packets to 14.04 packets.

For the cases where for source 2, $p_+ = 0.31$, $p_- = 0.29$, we, again, notice how using the proposed assignment algorithm helps keep the buffer occupancy low. The average occupancy decreases from 38.7 packets to 20.85.

Fig. 10 illustrates the buffer occupancy probabilities for several cases of five-source scenarios. The solid line represents the case where the delay models used for all sources is uniform (this is achieved by using $p_+ = p_-$) in the range [0, 10] time units.

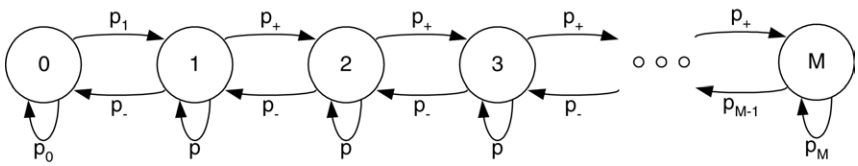

Fig. 8. The simple, finite Markov-chain representing the delay.

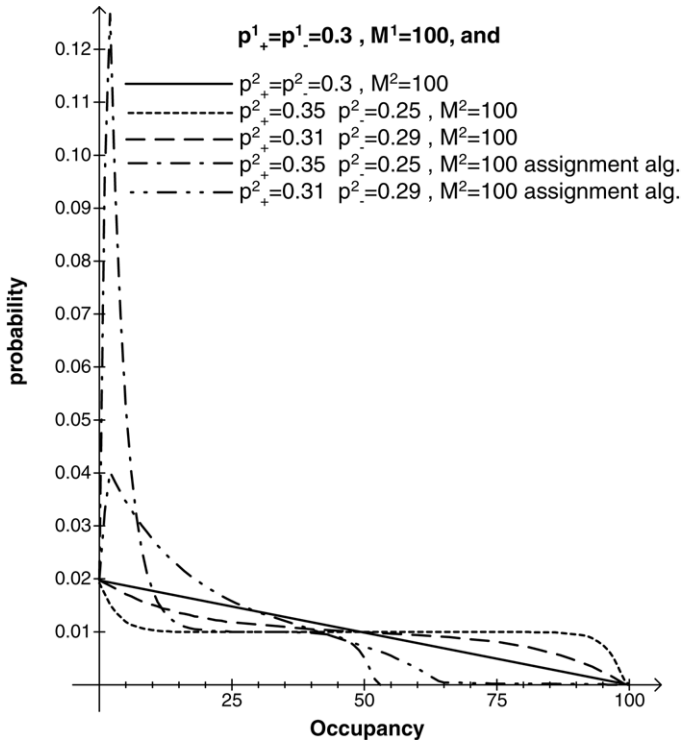

Fig. 9. Buffer occupancy probability—two sources.

For the cases where for sources 1 through 4 $p_+ = 0.16$, $p_- = 0.44$, we can notice how using the proposed assignment algorithm helps keep the buffer occupancy low. The average occupancy decreases from 14.27 packets to 7.68 packets.

The cyclic pattern that emerges when the proposed assignment algorithm is not applied is due to the fact that in this scenario the source of the mvp is source 5, with large probability, since the average delay of packets transmitted by this source is 5, while the average delay of packets transmitted by sources 1 through 4 is 1.007. When this occurs, packets stored in the resequencing-buffer are packets transmitted by sources 1 through 4. Each source contributes packets to the resequencing-buffer, corresponding to the difference between the delay from the source of the mvp and its own delay. Therefore, if, for instance, all four sources have the same delay, the resulting occupancy will be a multiple of 4, if three of the four sources have the same delay, and the delay from the fourth is one time unit larger, the occupancy will be of the form $4n + 1$, $n = 1, 2, 3, \ldots$. Consequently, the occupancy probability has a cyclic pattern, with cycle 4.

However, when the proposed assignment algorithm is applied, the probability of source 5 being the source of the mvp decreases significantly since this case is equivalent to the case where the average delay is the same for all the sources, thus the cyclic pattern disappears.

For the cases where for source 1, $p_+ = 0.16$, $p_- = 0.44$, we again notice how using the proposed assignment algorithm helps in keeping the buffer occupancy low. The average occupancy decreases from 17.04 packets to 13.79 packets. Evidently, the improvement is much less significant than in the former

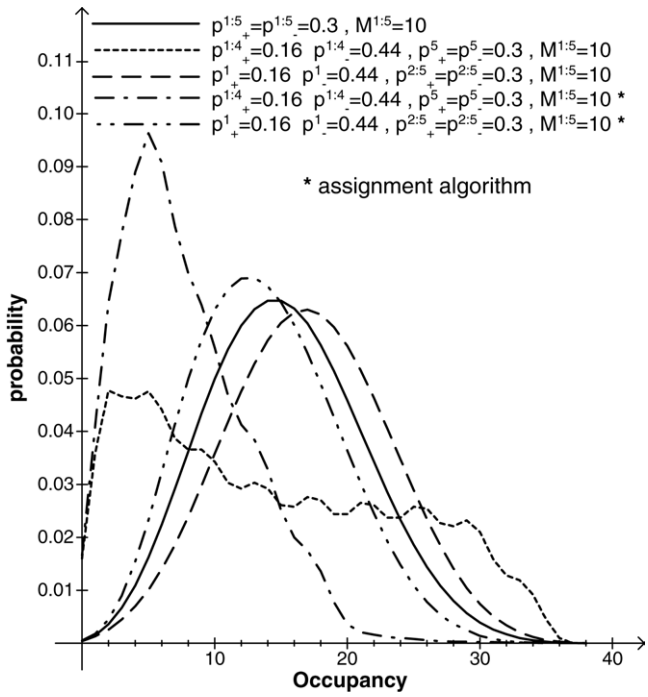

Fig. 10. Buffer occupancy probability—five sources.

case. This is because the new assignment effects only source 1 in this scenario, while in the former four sources where effected.

Fig. 11 illustrates the buffer occupancy probabilities for several scenarios where two sources transmit packets, with different packet rates. The TRR is the transmission rate ratio (i.e., if TRR = 3:5, source 1 transmits three packets in a global time unit, while source 2 transmits five packets in a global time unit).

The delay of packets transmitted by source 1 is uniformly distributed in [0, 100] source 1 time units, since $p_+ = p_-$. Therefore, the average delay of packets transmitted by source 1 is 50 source 1 time units.

For the case where for source 2 $p_+ = 0.295$, $p_- = 0.305$, the average delay of packets transmitted by source 2 is 26.2 source 2 time units. Therefore, when the TRR = 3:5, the average delay of packets transmitted by source 1 is $\frac{50}{3} = 16\frac{2}{3}$ global time units, whereas the average delay of packets transmitted by source 2 is $\frac{26.2}{5} = 5.24$ global time units. Consequently, the source of the mvp is source 1, with very large probability. On the other hand, when TRR = 5:3, the average delay of packets transmitted by source 1 is $\frac{50}{5} = 10$ global time units, whereas the average delay of packets transmitted by source 2 is $\frac{26.2}{3} = 8.73$ global time units. Consequently, the source of the mvp may be source 1 or source 2, with similar probabilities. Anyway, if the mvp was transmitted by source 1, the occupancy would have an upper limit of 60, since the maximal occupancy will be reached if the delay from source 1 was at its maximum, i.e. 100 source 1 time units, and the delay from source 2 was at its minimum, i.e. 0 source 2 time units. In this case the packets stored in the resequencing-buffer are all the packets transmitted by source 2 since the last packet received from source 1 was transmitted. Therefore, given that the delay of source 1 is at its maximum, i.e. $\frac{100}{5} = 20$ global time units, the maximum number of packets stored in the buffer is exactly $20 \times 3 = 60$ packets. This explains the dramatic slope change at occupancy 60, since

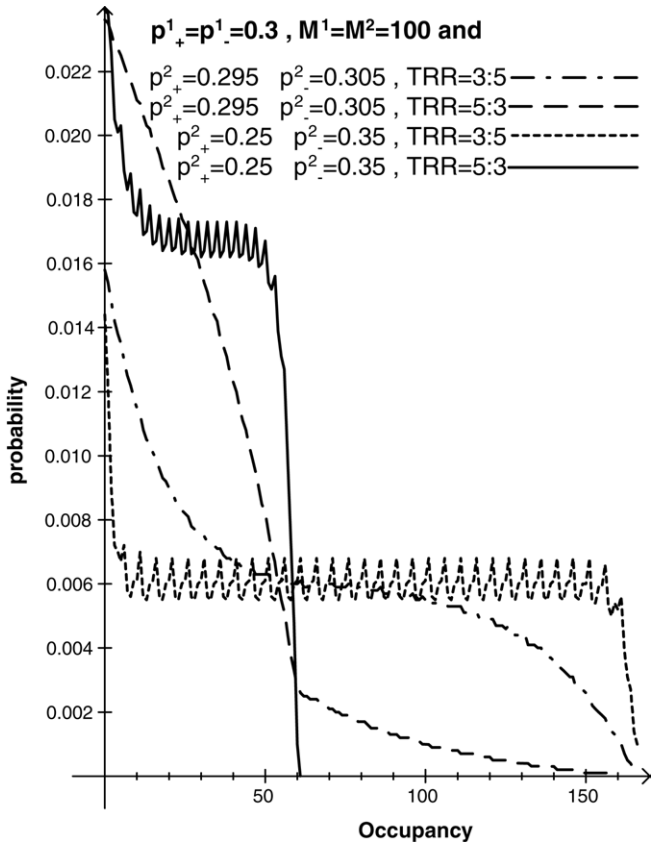

Fig. 11. Buffer occupancy probability—two sources with different transmission rates.

occupancy under 60 may be reached if the source of the mvp is either source 1 or 2, while occupancy over 60 may be reached only if the source of the mvp is source 2.

For the case where for source 2, $p_+ = 0.25$, $p_- = 0.35$, the average delay of packets transmitted by source 2 is 2.9 source 2 time units. Therefore, when TRR = 3:5, the average delay of packets transmitted by source 1 is $\frac{50}{3} = 16\frac{2}{3}$ global time units, while the average delay of packets transmitted by source 2 is $\frac{2.9}{5} = 0.58$ global time units. Consequently, the source of the mvp is source 1, with very large probability.

When TRR = 5:3, the average delay of packets transmitted by source 1 is $\frac{50}{5} = 10$ global time units, whereas the average delay of packets transmitted by source 2 is $\frac{2.9}{3} = 0.97$ global time units. Consequently, the source of the mvp is, again, source 1, with very large probability. As explained earlier, when TRR = 5:3, and the source of the mvp is source 1, maximal occupancy is 60, which explains the dramatic drop in the occupancy probability near occupancy 60.

In both TRR cases a cyclic pattern is noticed. This pattern is a result of having a very low variance in the delay of packets transmitted by source 2, which is the source contributing to the buffer occupancy. When TRR = 3:5, 5 packets are transmitted by source 2 every time unit, therefore, the cycle period is five packets, whereas when TRR = 5:3, three packets are transmitted by source 2 every time unit, therefore the cycle period is three packets.

7. Summary and discussion

This paper analyzes the approach of receiving data packets of resources in parallel from several servers, where the resource is available identically, through requesting transmission of specific portions of the resource from each relevant server. The model assumed a constant packet rate from each source, and a stationary delay model, where packets received from each source arrive in the order transmitted.

Closed form equations for calculating the resequencing-buffer occupancy distribution while downloading a resource in parallel from several sources were presented for isochronous sources and for sources with different transmission rates.

Streaming applications that require a reliable constant bit rate may take advantage of the robustness that the source diversity offers. In such applications it is desirable to minimize the resequencing-buffer demands and achieve a more regulated flow of data to the application at the destination, this allows for shorter playback delays and lowers the probability of interruption at the application.

An algorithm for assigning packets to be transmitted from each source was proposed. The performance of the algorithm was analyzed. Numerical results show improved performance, especially when the delays of packets transmitted by the different sources are likely to differ significantly. The algorithm presented here improves on the buffer occupancy based on static knowledge of the average delay. A more realistic algorithm should take advantage of temporal measurement of the delay, and dynamically adapt its assignment strategy to the ever changing network conditions. It is straightforward to incorporate such dynamic adjustments to the data requests while attempting to obey the assignment criterion.

The analysis in this paper assumes fine granularity packet level requests. When the network conditions change rapidly, accurate estimation of the average delay may be difficult to come by and long periods of averaging may be required. Also, one may wish to simplify the scheme by lowering the frequency of requests from the user to the sources. Therefore, a scheme in which chunks of data are requested from the sources may be adopted. In such a case the sizes of the requested chunks should be optimized.

Questions for future work, regarding the analysis of this approach, include investigating other delay models. The model analyzed here accounts for packet loss and out of order arrival of packets from a single source only if out of order packets from any single source are assumed to be dropped. Another problem is to model a case where the paths from the sources to the destination are not bottleneck disjoint, which results in dependence in the delays of packets transmitted by the different sources. It would be interesting to test a dynamic version of the proposed assignment algorithm that uses temporal measurements of the delay, while taking the accuracy of the measurements into account when deciding on the sizes of data chunks that should be requested from each source. In such a case measurements of the duration required to complete downloads of chunks of data rather than packet delay may be of interest. Based on this information, instead of attempting to compensate for delay differences by adjusting the packet assignments, one may dynamically adjust the sizes of the requested data chunks to equalize the download duration from the different sources.

Acknowledgment

We would like to thank the anonymous reviewers for their insightful and constructive comments and suggestions, helping us in simplifying some of the proofs and improving readability and clarity of the paper.

Appendix A

Proof of Theorem 5. We assume the instance we observe the resequencing-buffer occupancy is t (global) time units.

The last packets that arrived from sources 1 and 2 were transmitted at times $t-\frac{\delta_1}{a}$ and $t-\frac{\delta_2}{b}$, respectively. The mvp was transmitted at time $\min\{t-\frac{\delta_1-1}{a}, t-\frac{\delta_2-1}{b}\}$. Packets stored in the resequencing-buffer are packets that have arrived, and were transmitted by the source that did not transmit the mvp after the mvp was transmitted.

The mvp was transmitted by source 2 if $\min\{t-\frac{\delta_1-1}{a}, t-\frac{\delta_2-1}{b}\} = t-\frac{\delta_2-1}{b}$, and $t-\frac{\delta_1-1}{a} \neq t-\frac{\delta_2-1}{b}$. This occurs if and only if $a\delta_2 - b\delta_1 > a-b$. In this case all packets transmitted by source 1 in the time interval $(t-\frac{\delta_2-1}{b}, t-\frac{\delta_1}{a}]$ are stored in the resequencing-buffer. Therefore, for $a\delta_2 - b\delta_1 > a-b$, the number of packets stored in the resequencing-buffer is

$$B = \left\lceil \left[\left(t-\frac{\delta_1}{a}\right) - \left(t-\frac{\delta_2-1}{b}\right) \right] a \right\rceil = \left\lceil \frac{(\delta_2-1)a}{b} \right\rceil - \delta_1 \tag{21}$$

The mvp was transmitted by source 1 if $\min\{t-\frac{\delta_1-1}{a}, t-\frac{\delta_2-1}{b}\} = t-\frac{\delta_1-1}{a}$, even if $t-\frac{\delta_1-1}{a} = t-\frac{\delta_2-1}{b}$. This occurs if and only if $a\delta_2 - b\delta_1 \leq a-b$. In this case all packets transmitted by source 2 in the time interval $[t-\frac{\delta_1-1}{a}, t-\frac{\delta_2}{b}]$ are stored in the resequencing-buffer. Therefore, for $a\delta_2 - b\delta_1 \leq a-b$, the number of packets stored in the resequencing-buffer is

$$B = \left\lfloor \left[\left(t-\frac{\delta_2}{b}\right) - \left(t-\frac{\delta_1-1}{a}\right) \right] b + 1 \right\rfloor = \left\lfloor \frac{(\delta_1-1)b}{a} + 1 \right\rfloor - \delta_2 \tag{22}$$

From (21) and (22) we conclude that (18) holds.

References

[1] Z. Fei, S. Bhattacharjee, E.W. Zegura, M.H. Ammar, A Novel Server Selection Technique for Improving the Response Time of a Replicated Service, INFOCOM'98, pp. 783–791.
[2] R.L. Carter, M.E. Crovella, Server Selection Using Dynamic Path Characterization in Wide-Area Networks, INFOCOM'97, pp. 1014–1021.
[3] M. Sayal, Y. Breitbart, P. Scheuermann, R. Vingralek, Selection Algorithms for Replicated Web Servers, Workshop on Internet Server Performance Madison, Wisconsin, 1998.
[4] S. Seshan, M. Stemm, R.H. Katz, SPAND: shared passive network performance discovery, in: Proceedings of USENIX Symposium on Internet Technologies and Systems, Monterey, CA, 1997.
[5] J.W. Byers, M. Luby, M. Mitzenmacher, Accessing Multiple Mirror Sites in Parallel: Using Tornado Codes to Speedup Downloads, INFOCOM'99, pp. 275–283.
[6] M. Luby, M. Mitzenmacher, A. Shokrollahi, D. Spielman, V. Stemann, Practical loss-resilient codes', in: Proceedings of the 29th annual ACM Symposium on Theory of Computing, 1997, pp. 150–159.
[7] L. Rizzo, Effective erasure codes for reliable computer communication protocols, Computer Commun. Rev. 27 (2) (April 1997) 24–36.
[8] J. Byers, M. Luby, M. Mitzenmacher, A. Rege, A digital fountain approach to reliable distribution of bulk data, in: Proceedings of ACM SIGCOMM, 1998, pp. 56–67.
[9] J. Byers, J. Considine, M. Mitzenmacher, S. Rost, Informed Content Delivery Across Adaptive Overlay Networks, SIGCOMM'02, pp. 47–60.

[10] K.C. Almeroth, M.H. Ammar, Z. Fei, Scalable Delivery of Web Pages Using Cyclic Best-Effort Multicast, INFOCOM'98, pp. 1214–1221.
[11] A. Jean-Marie, M. Tidball, M. Escalante, V. Leoni, H. Ponce de Leon, On the influence of resequencing on the regularity of service, Perform. Eval. J. 36–37 (1999) 115–135.
[12] P. Rodriguez, E.W. Biersack, Dynamic parallel access to replicated content in the Internet, IEEE/ACM Trans. Netw. 10 (4) (August 2002) 455–465.
[13] Y. Nebat, M. Sidi, Analysis of resequencing in downloads, Int. J. Commun. Syst. 16 (2003) 735–757.
[14] A.G. Pakes, Some conditions for ergodicity and recurrence of Markov chains, Operations Res. 17 (1969) 1058–1061.
[15] F. Baccelli, A.M. Makowski, Queueing models for systems with synchronization constraints, Proc. IEEE 77 (1) (1989) 138–161.
[16] Z. Rosberg, N. Shacham, Resequencing delay and buffer occupancy under the selective-repeat ARQ, IEEE Trans. Inform. Theory 35 (1) (1989) 166–173.
[17] J. Berster, Recent results in Sturmian words, in: J. Dassow, A. Salomaa (Eds.), Development in Language Theory, World Scientific, Singapore, 1996, pp. 13–24.
[18] Y. Nebat, M. Sidi, Resequencing Considerations in Parallel Downloads, CCIT Report #348, Electrical Engineering Dept., Technion, Haifa, Israel, July 2001.

Yoav Nebat received the BSc and MS degrees in electrical engineering from the Technion-Israel Institute of Technology, Haifa, Israel in 1995 and 2002, respectively. Mr. Nebat is currently pursuing the PhD degree in electrical engineering at the University of California, San Diego. His current research interests are in cross-layer design, signal processing and information theoretic analysis for wireless multi-user communication systems.

Moshe Sidi received the BSc, MSc and the DSc degrees from the Technion-Israel Institute of Technology, Haifa, Israel, in 1975, 1979 and 1982, respectively, all in electrical engineering. In 1982 he joined the faculty of Electrical Engineering Department at the Technion where he is currently a Professor holding the Technion Chair for Electrical Engineering. During the academic year 1983–1984 he was a Post-Doctoral Associate at the Electrical Engineering and Computer Science Department at the Massachusetts Institute of Technology, Cambridge, MA. During 1986–1987 he was a visiting scientist at IBM, Thomas J. Watson Research Center, Yorktown Heights, NY. He coauthors the book "Multiple Access Protocols: Performance and Analysis," Springer Verlag 1990. He served as the Editor for Communication Networks in the IEEE Transactions on Communications from 1989 until 1993, as the Associate Editor for Communication Networks and Computer Networks in the IEEE Transactions on Information Theory from 1991 until 1994, as an Editor in the IEEE/ACM Transactions on Networking from 1993 until 1997, and as an Editor in the Wireless Journal 1993–2001. He also served as the General Chair for Infocom 2000. He was awarded the 1983–1984 Rothschild Fellowship for postdoctoral research, the 1983–1984 Fulbright Award for postdoctoral research, the 1985–1986 Bat-Sheva De Rothschild Fund for Young Distinguished Researchers, the 1989–1990 New England Academic Award and the 1997 Muriel and David Jacknow Award for Excellence in Teaching. His research interests are in wireless networks and multiple access protocols, traffic characterization and guaranteed grade of service in high-speed networks, queueing modeling and performance evaluation of computer communication networks. He published more than 140 papers in these areas in leading journals and conferences.