Noname manuscript No.
(will be inserted by the editor)

Delay and playout probability trade-off in mesh-based peer-to-peer streaming with delayed buffer map updates

Ilias Chatzidrossos · György Dán · Viktoria Fodor

the date of receipt and acceptance should be inserted later

Abstract In mesh-based peer-to-peer streaming systems data is distributed among the peers according to local scheduling decisions. The local decisions affect how packets get distributed in the mesh, the probability of duplicates and consequently, the probability of timely data delivery. In this paper we propose an analytic framework that allows the evaluation of scheduling algorithms. We consider four solutions in which scheduling is performed at the forwarding peer, based on the knowledge of the playout buffer content at the neighbors. We evaluate the effectiveness of the solutions in terms of the probability that a peer can play out a packet versus the playback delay, the sensitivity of the solutions to the accuracy of the knowledge of the neighbors' playout buffer contents, and the scalability of the solutions with respect to the size of the overlay. We also show how the model can be used to evaluate the effects of node arrivals and departures on the overlay's performance.

1 Introduction

Recent years have seen the proliferation of peer-to-peer (P2P) systems for the delivery of live streaming content on the Internet [1–3]. These systems are popular because they allow content providers to save on infrastructure costs: users contribute with their upload bandwidth resources to the distribution of the content. Whether P2P streaming systems will become an alternative to commercial, operator managed IPTV solutions depends on whether they can offer a comparable user experience.

The user experience in a streaming system is typically determined by two interdependent performance measures [4]: the playback delay and the playout probability. The playback delay determines how real-time streaming can be, but also the zapping times, that is, the delays experienced when users switch channel. The playout probability is the ratio of packets successfully played out at the receiver. Additionally, the performance of a P2P streaming system can be evaluated with respect to two other metrics. First, the playback lag between the peers has to be low to provide similar real-time experience to all users. Second, the performance in terms of the previous three metrics has to scale well with the overlay's size.

P2P streaming systems proposed in the literature generally fall into one of two categories: multiple-tree based and mesh-based solutions [5]. Mesh-based solutions are considered to be more robust to dynamic overlay membership and fluctuating transmission capacities between the nodes, hence they are prevalent in commercial systems. Nevertheless as a consequence of their dynamic nature, there is a limited analytical understanding of the performance trade-offs of mesh-based solutions.

In a mesh-based system neighbor relations between the peers are determined locally, and consequently, there is no global structure maintained. The forwarding of data packets (or often a set of data packets) is decided locally at the peers, either on the transmitting or on the receiving side. As shown in Figure 1.a, the packet scheduling is based on information exchanged between the neighboring peers: peers report the packets they already possess, with the so called buffer map. Clearly, scheduling algorithms can be more efficient if they uti-

ACCESS Linnaeus Centre, School of Electrical Engineering
KTH, Royal Institute of Technology
Osquldas väg 10, 100-44, Stockholm, Sweden
E-mail: iliasc@ee.kth.se, gyuri@ee.kth.se, vfodor@ee.kth.se

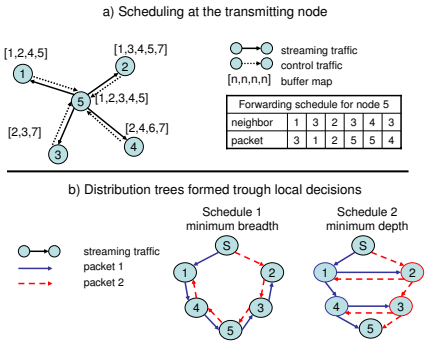

Fig. 1 Mesh based streaming with local scheduling decision. a) Forwarding node 5 schedules neighbors and packets based on the received buffer maps. b) The distribution trees are formed through local decisions.

lize detailed and timely information, and therefore the trade-off of streaming performance and control overhead has to be considered.

As a result of local scheduling decisions, the packets of the stream will reach the peers via different trees over the mesh. We call these trees the distribution trees. The structure of the distribution trees is determined by the local decisions at the peers. Figure 1.b shows two examples where the distribution trees are minimum breadth and minimum depth trees respectively, with a depth of 5 and 3 overlay hops. The challenge of modelling mesh-based streaming comes then from the following closed loop: on one side the scheduling decisions determine the distribution trees formed over the mesh, and thus the availability of the packets at the peers, on the other side, the packet availability at the peers affects the scheduling decisions that can be made.

In this paper we propose an approach to cut this loop. We assume that the peers' positions in the distribution trees are statistically the same, and consequently, the probability that a peer receives a packet with given delay is the same for all peers. We validate our assumption with simulation and build then an analytic model to describe the packet reception process. We propose an analytic framework that can be used to evaluate the performance of different scheduling strategies. In this paper we consider four specific scheduling strategies, where scheduling is performed at the transmitting peers. We evaluate how the playout probability is affected by the number of neighbors, the playback delay and the frequency of buffer map updates. Then we evaluate the scalability of the solutions in terms of overlay size and derive conclusions on the structure of the distribution trees. We also use the model to evaluate the effects of node churn on the overlay performance.

The rest of the paper is organized as follows. In Section 2 we overview related work. Section 3 describes the P2P streaming systems we consider in this paper. We give the analytic model in Section 4, and evaluate the performance of the considered solutions in Section 5. Section 6 concludes our work.

2 Related Work

Several measurement studies deal with the analysis of commercial, mesh-based P2P streaming systems [6–8]. The goal of these studies is to understand the proprietary protocol used by the streaming system or to understand the effects of the networking environment and the user behavior. There are also several works in which solutions for mesh-based streaming are proposed and evaluated through simulations [9–14]. These works often consider a different set of input parameters and performance measures, and therefore the proposed systems are hard to compare. There are however few works that address the streaming performance with analytic models. In [15] the minimum playback delay is derived for given source and peer transmission capacities, by considering the trade-off between the depth of the distribution trees and the time a peer needs to forward a packet to all of its downstream neighbors. The performance of BitTorrent-based streaming is addressed in [16], deriving relation between the achievable utilization of upload bandwidth and the number of segments or packets considered for sharing, which in turn is related to the playback delay peers will experience. Rate-optimal streaming in complete meshes is considered in [17] using a queueing theoretic approach. The authors show that rate-optimal streaming is possible in a distributed manner and derive scheduling algorithms for the cases when the bottleneck is at the uplink or at the downlink of the peers. This work is extended in [18] where the authors propose different algorithms, and evaluate their delay-throughput performance in overlays forming a complete graph, using mean-field approximations. We extended this work by relaxing the assumption on complete overlay graph in [20]. We derived analytical expressions for the data propagation in a mesh-based overlay with random packet forwarding given that peers always have perfect knowledge of the buffer contents of their neighbors. The present work extends [20] by capturing the effect of outdated buffer map information at the sender peer, by considering a larger set of forwarding schemes and by including the case of overlays with node churn.

The scalability properties of P2P streaming systems are addressed in [19] by use of large deviations theory.

The paper proves for a large set of node-to-node delay distributions that in order to keep the same playout probability, the playback delay has to be increased in proportion to the increase of the overlay path lengths. We use this result to assess the structure of the distribution trees, even if they are formed in a distributed manner through the local decisions of the transmitting peers.

3 System overview

We consider a peer-to-peer live streaming system consisting of a streaming server and N peers. The peers form an overlay such that each peer knows about up to d other peers, called neighbors, and about the streaming server. In the case of node churn, arriving peers become neighbors of d peers that have less than d neighbors. We assume that once a peer leaves the overlay its neighbors do not actively look for new connections but wait until a newly joining peer connects to them.

The system is time slotted, with one slot being equal to the packet content duration. At the beginning of each time slot a new packet is generated at the server and is forwarded to m randomly chosen peers. Peers distribute the data to their neighbors according to some scheduling algorithm. At the beginning of every time-slot every peer makes a forwarding decision, i.e., chooses a neighbor and a packet, and sends the chosen packet to the chosen neighbor. Packets that are sent at the beginning of a time-slot reach their destination at the end of the same slot. The upload capacity of peers is equal to the stream rate, i.e., one packet per slot, while their download capacity is unlimited, i.e., they can receive up to d packets from their neighbors in one slot.

Each peer maintains a playout buffer of B packets in which it stores the packets received from its neighbors before playing them out. A peer that joins the overlay before the beginning of time slot i starts playout with packet i at the beginning of time slot $i + B$. Consequently, the playback is synchronized among all peers, and both the startup delay and the playback delay are B time slots.

The nodes' decisions on forwarding a packet are based on the information they have about the packets that their neighbors possess in their playout buffers. Nodes know about the contents of their neighbors' playout buffers via exchanging buffer maps. In a scenario with *perfect information*, each peer would have full knowledge of the buffer contents of its neighbors upon taking a forwarding decision. However, such a scheme would be infeasible in a real peer-to-peer system due to the overhead related to the buffer map exchange among all the peers in the overlay. Consequently, we consider that nodes send updated buffer maps to their neighbors every u time slots, and hence, forwarding decisions are made based on *outdated information*. The shorter the update interval, the more accurate is the information in the buffer maps, but at the same time the higher is the overhead due to message exchange in the overlay.

3.1 Push-based forwarding schemes

In this paper we consider four push-based forwarding schemes. All schemes take as input the buffer maps of all the neighbors and return as output the forwarding decision, i.e., a (neighbor, packet) pair. Before describing the forwarding schemes, we define the notion of eligible packet and that of eligible neighbor. We say that a packet is eligible with respect to a neighbor if the peer has the packet in its playout buffer but the packet is not present in the most recently received buffer map from the neighboring peer. Similarly, we say that a neighbor of a peer is eligible if the peer has at least one eligible packet with respect to that neighbor.

Random peer - Random packet (RpRp): The peer constructs the set of eligible neighbors based on the buffer maps received from its neighbors, and picks uniformly at random one neighbor from the set. The peer then creates the set of eligible packets with respect to the chosen neighbor, and picks a packet uniformly at random from the set.

Determined peer - Random packet (DpRp): The peer calculates the number of eligible packets for all of its eligible neighbors based on the buffer maps. It selects a neighbor with a probability proportional to the number of eligible packets with respect to that neighbor. The packet to be sent to the selected neighbor is then chosen uniformly at random from the set of eligible packets with respect to that neighbor. By following this scheme a peer forwards data with higher probability to a neighbor to which it has many eligible packets.

Latest blind packet - Random useful peer (LbRu): The peer chooses the packet with the highest sequence number in its playout buffer independent of whether the packet is eligible with respect to any of its neighbors. It then chooses uniformly at random one of its neighbors that are missing the chosen packet. The unconditional selection of the packet implies that there might be cases when a peer cannot forward the last packet because all the neighbors already have it, while some neighbor might be missing a packet with lower sequence number that the forwarding peer has.

Latest useful packet - Useful peer (LuUp): The peer chooses the packet that has the highest sequence number among the packets that are eligible with

respect to at least one neighbor. It then picks uniformly at random among the neighbors that are eligible with respect to the chosen packet. The chosen packet is not necessarily the last packet in the buffer, which is the difference of this scheme compared to the *LbRu* forwarding scheme.

We will refer to the two first schemes as random-packet schemes as they pick the packet to be forwarded at random, and we refer to the two latter schemes as fresh-data-first schemes because they always try to forward the data with the highest sequence numbers. *RpRp* was previously evaluated for the specific case of per slot buffer map exchange [20] and fresh-data-first schemes were analyzed in [18] for the case when the overlay forms a complete graph and with the assumption that local scheduling decisions at the transmitting peers are immediately known in the neighborhood of the receiving peer, that is, transmission of duplicates is fully avoided.

4 Analytical Model - Data dissemination

In this section we first describe the analytical model of an overlay without churn, then we show how this model can be used to approximate the effects of node churn. While the model considers a slot synchronized system, we argue that it describes well also the behavior of unsynchronized systems, referring to results in [20].

The key assumption of our model is that the contents of the playout buffers of neighboring peers are independent and statistically identical. The assumption is based on the observation that if peers maintain a fair number of neighbors then the local forwarding decisions will lead to per packet distribution trees that are very different, and as a result the position of the peers in the distribution trees is statistically the same. We validate this assumption by simulations in Section 5.

We number packets according to their transmission time at the server, i.e., packet j is transmitted in slot j. Packet j is played out at the beginning of slot $j+B$ at each peer. Since playout occurs at the beginning of the time slots, a peer can play out packet j if the packet is in the playout buffer of the peer by the end of time slot $j+B-1$.

Let us denote by $\ell(i)$ the lowest packet sequence number that peers would potentially forward in time slot i. Since in slot i a peer would not forward the packet that is to be played out in that slot, $\ell(i) = max\{0, i-B+1\}$. The highest packet sequence number that can be forwarded by any peer in slot i is $i-1$, because only the source has packet i in slot i.

We denote by $\mathcal{B}_i^r \in 2^{\{\ell(i),\ldots,i-1\}}$ the set of packets that peer r has in its playout buffer at the beginning of slot i, i.e., the set of packets that it could potentially forward in slot i. In Figure 2 we show an example of a peer's playout buffer at the beginning of three consecutive time slots. The packet in the bold cell is the packet that should be played out at the given slot. At the beginning of slot $i=9$ the playout of the packets has not started and all the packets present in the buffer belong to the set $\mathcal{B}_i$. At the beginning of slot $i=10$, packet 0 is to be played out, so it cannot be eligible for sending and the same holds for packet 1 at time $i=11$.

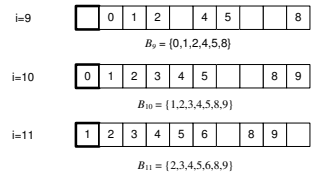

Fig. 2 Example of a playout buffer of a peer and the set of packets that the peer could forward for three consecutive time-slots. Cell with bold edges contains the packet that is to be played out during the specific slot. In slot i=9 the peer receives packet 3 from a neighbor and packet 9 from the source. In slot i=10 it receives packet 6 from a neighbor.

Let us denote by P_i^j the probability that a peer is in possession of packet j by the end of time-slot i, i.e., $P(j \in \mathcal{B}_i) = P_{i-1}^j$. Hence, packet j will be successfully played out with probability P_{j+B-1}^j. Similarly, let us denote by $\overline{P}_i^j$ the probability that a peer possesses packet j at the end of slot i as known by one of its neighbors at the end of slot i. Since buffer map updates do not occur instantaneously, $\overline{P}_i^j = P_i^j$ does not necessarily hold. We consider that buffer map updates occur every u time slots, with the first exchange at the end of slot $i=0$, so that we can express $\overline{P}_i^j$ as a function of P_i^j

$$\overline{P}_i^j = \begin{cases} 0 & u = \infty \\ P_{\lfloor i/u \rfloor u}^j & 1 \le u < \infty. \end{cases} \quad (1)$$

In the following we derive recursive formulae to calculate the probability of successful playout in an overlay overlay without and with node churn.

4.1 Overlay without node churn

Let us consider an overlay in which peers do not join nor leave and there are no losses in the overlay links between peers. Our goal is to calculate the probability P_i^j for an arbitrary peer r in the overlay. The peer has packet j at the end of slot i either if it already had it at the end of slot $i-1$, or if it did not have it at the end of slot $i-1$ but received it from some neighbor s during slot i.

This can be expressed by a general recursive equation introduced in [20]

$$P_i^j = \begin{cases} 0 & i < j \\ \frac{m}{N} & i = j \\ P_{i-1}^j + (1 - P_{i-1}^j)(1 - (1 - \pi_i^j)^d) & i > j. \end{cases} \quad (2)$$

where π_i^j is the probability that some neighbor s forwards packet j in slot i to peer r, given that peer r does not have packet j. P_j^j is the probability that a node receives the packet directly from the server. P_i^j is determined by π_i^j, i.e, it depends on the forwarding scheme used. In the following subsections we will derive expressions for the π_i^j for the *RpRp*, *DpRp*, *LbRu* and *LuUp* forwarding schemes. When describing the four schemes we refer to the peer that makes the forwarding decision as peer s, and to the peer that is to receive packet j as peer r.

4.1.1 Random peer - Random packet

In the *RpRp* scheme peer s first chooses a peer, and then it chooses a packet to be sent. Let us define the corresponding events

$A := \{s \text{ has packet } j \text{ and } r \text{ does not have it}\}$,
$C := \{s \text{ chooses to send a packet to } r\}$,
$D := \{s \text{ chooses to send packet } j\}$.

Using these events the probability π_i^j can be written as

$$\begin{aligned} \pi_i^j &= P(s \text{ sends packet } j \text{ to } r | j \notin \mathcal{B}_i^r) \\ &= P(s \text{ sends packet } j \text{ to } r | j \notin \mathcal{B}_i^r, j \in \mathcal{B}_i^s) \cdot P(j \in \mathcal{B}_i^s) \\ &= P(C|A) \cdot P(D|A \cdot C) \cdot P(j \in \mathcal{B}_i^s). \end{aligned} \quad (3)$$

By definition $P(j \in \mathcal{B}_i^s) = P_{i-1}^j$, so we proceed to the calculation of $P(C|A)$. Under the *RpRp* scheme, peer s transmits to one of its eligible neighbors. Given the condition that node r is eligible, the probability that node s selects node r to transmit to is

$$P(C|A) = \sum_{k=0}^{d-1} \frac{1}{k+1} \binom{d-1}{k} p_e^k (1 - p_e)^{d-k-1}, \quad (4)$$

where p_e is the probability that a neighbor is eligible

$$p_e = (1 - \overline{P}_{i-1}^j) + \overline{P}_{i-1}^j (1 - \prod_{w=\ell(i), w \neq j}^{i-1} (1 - P_{i-1}^w (1 - \overline{P}_{i-1}^w))). \quad (5)$$

Finally, we calculate $P(D|A \cdot C)$, the probability that s will send packet j and not another eligible packet to r. Let us denote by the r.v. K the number of packets that are eligible to be sent from s to r including packet j. Clearly, $K \geq 1$ because packet j is eligible with respect to node r. Then the probability that packet j is sent is

$$P(D|A \cdot C) = \sum_{k=1}^{i-\ell(i)} \frac{1}{k} P(K = k). \quad (6)$$

In the following define a recursion to calculate the distribution of K.

We define the probability vector

$$Q = \{P_i^{\ell(i)} \cdots P_i^{j-1}, 1, P_i^{j+1} \cdots P_i^{i-1}\},$$

which contains the probabilities that node s has packets at the different buffer positions, given that it has packet j. Similarly, we define the probability vector

$$\overline{Q}_r = \{\overline{P}_i^{\ell(i)} \cdots \overline{P}_i^{j-1}, 0, \overline{P}_i^{j+1} \cdots \overline{P}_i^{i-1}\},$$

which represents the information that the sending peer s has about the buffer contents of peer r, given that peer r does not have packet j.

We initialize the recursion by setting $L_{0,0} = 1$ and $L_{k,0} = 0$ for $k < 0$. The recursion is then defined by

$$\begin{aligned} L_{k,l} = L_{k,l-1}(1 - Q(l)(1 - \overline{Q}_r(l))) + \\ L_{k-1,l-1} Q(l)(1 - \overline{Q}_r(l)). \end{aligned} \quad (7)$$

The recursion is executed for $l = 1 \ldots i - \ell(i)$ and for every l for $k = 0 \ldots l$. The distribution of K is then given as $P(K = k) = L_{k,i-\ell(i)}$ for for $k = 1 \ldots i - \ell(i)$. By plugging (4) and (6) into (3) we obtain the probability π_i^j.

4.1.2 Determined peer - Random packet

Similar to the *RpRp* forwarding scheme, and using the events introduced there, we can write

$$\begin{aligned} \pi_i^j &= P(s \text{ sends packet } j \text{ to } r | j \notin \mathcal{B}_i^r) \\ &= P(s \text{ sends packet } j \text{ to } r | A) \cdot P(j \in \mathcal{B}_i^s). \end{aligned} \quad (8)$$

Nevertheless, for the *DpRp* forwarding scheme the choice of a peer and the choice of a packet are correlated. Hence we calculate the joint probabilities directly.

Let us denote by the r.v. K the total number of forwarding decisions that node s can make with respect to any of its neighbors (i.e., node-packet pairs), given that s possesses j and r does not possess j. Clearly, $K \geq 1$ because packet j is eligible with respect to node r. Given the distribution of K the conditional probability that packet j is sent to r is expressed as

$$P(s \text{ sends packet } j \text{ to } r | A) = \sum_{k=1}^{d(i-\ell(i))} \frac{1}{k} P(K = k). \quad (9)$$

In the following we define a recursion to calculate the distribution of K. Apart from the vectors Q and $\overline{Q}_r$ defined in the previous section, we define the probability vector

$$\overline{Q} = \{\overline{P}_i^{\ell(i)} \cdots \overline{P}_i^{i-1}\},$$

which represents the information that the sending peer s has about the buffer contents of its neighboring peers apart from node r.

As before, we initialize the recursion by setting $L_{0,0} = 1$ and $L_{k,0} = 0$ for $k \neq 0$. The recursion is however defined by

$$\begin{aligned} L_{k,l} &= L_{k,l-1}\left\{(1-Q(l)) + Q(l)\overline{Q}(l)^{d-1}\overline{Q}_r(l)\right\} \\ &+ \sum_{z=1}^{d-1} L_{k-z,l-1}Q(l)\left\{\binom{d-1}{z}\overline{Q}(l)^{d-1-z}(1-\overline{Q}(l))^z\overline{Q}_r(l)\right. \\ &\left. + \binom{d-1}{z-1}\overline{Q}(l)^{d-z}(1-\overline{Q}(l))^{z-1}(1-\overline{Q}_r(l))\right\} \\ &+ L_{k-d,l-1}\left\{Q(l)(1-\overline{Q}(l))^d(1-\overline{Q}_r(l))\right\}. \end{aligned} \quad (10)$$

The recursion is executed for $l = 1, \dots i - \ell(i)$, and for every l for $k = 0 \dots ld$. The distribution of K is then given by $P(K = k) = L_{k,i-\ell(i)}$ for $k = 1 \dots (i - \ell(i))d$.

4.1.3 Latest blind packet - Random useful peer

The probability that r will receive packet j from s in time slot i is equal to the probability that j is the latest packet in the buffer of s, times the probability that s will choose r among its neighbors that do not have packet j. We define the events

$E := \{j \text{ is the latest packet in the buffer of } s\}$, and
$F := \{s \text{ chooses to send packet } j \text{ to } r\ \}$.

The expression for π_i^j can then be written as

$$\pi_i^j = P(E) \cdot P(\mathrm{F}|\mathrm{E} \cdot \mathrm{A}) \cdot P(j \in \mathcal{B}_i^s). \quad (11)$$

Again we have $P(j \in \mathcal{B}_i^s) = P_{i-1}^j$, and proceed to the calculation of $P(E)$. Event E occurs if there are no packets in the buffer of s that have a higher sequence number than j

$$P(E) = \prod_{v=j+1}^{i-1} P_{i-1}^v. \quad (12)$$

Finally, we calculate the probability $P(\mathrm{F}|\mathrm{E} \cdot \mathrm{A})$. We define the r.v. K as the number of neighbors of s other than r that are missing packet j. Then, we get that

$$\begin{aligned} P(F|E \cdot A) &= \sum_{k=0}^{d-1} \frac{1}{k+1} \cdot P(K = k) \\ &= \sum_{k=0}^{d-1} \frac{1}{k+1} \cdot \binom{d-1}{k}(1-(\overline{P}_i^j))^k \cdot (\overline{P}_i^j)^{d-1-k}. \end{aligned} \quad (13)$$

Combining the above equations we get the probability π_i^j.

4.1.4 Latest useful packet - Random useful peer

In addition to the events defined in the previous subsections let us define the event

$G := \{j \text{ is the latest useful packet in the buffer of } s\}$.

Using the same rationale as in the previous case, we can express π_i^j as

$$\begin{aligned} \pi_i^j &= P(s \text{ sends pkt } j \text{ to } r | j \notin \mathcal{B}_i^r) \\ &= P(s \text{ sends pkt } j \text{ to } r | A) \cdot P(j \in \mathcal{B}_i^s) \\ &= P(C|A \cdot G) \cdot P(G) \cdot P(j \in \mathcal{B}_i^s). \end{aligned} \quad (14)$$

Again, we have $P(j \in \mathcal{B}_i^s) = P_{i-1}^j$, so we turn to the probabilities $P(G)$ and $P(D|A \cdot G)$. Packet j is latest useful if there is no packet with higher sequence number that some of the neighbors of s are in need of

$$P(G) = \prod_{v=j+1}^{i-1} (1 - P_{i-1}^v(1 - (\overline{P}_{i-1}^v)^d)). \quad (15)$$

Then we calculate the probability $P(C|A \cdot G)$, that s will choose r to send the packet to, among all its neighbors

$$P(C|G \cdot A) = \sum_{k=0}^{d-1} \frac{1}{k+1} \cdot \binom{d-1}{k}(1-(\overline{P}_i^j))^k \cdot (\overline{P}_i^j)^{d-1-k}. \quad (16)$$

Plugging equations (15) and (16) into (14) yields the probability π_i^j. Note that the complexity of the presented models does not depend on the overlay size N, and therefore they provide excellent tools to evaluate the scalability of the scheduling schemes.

4.2 Overlays in the presence of node churn

Node-churn can affect the performance of a push-based system in three ways. First, if a peer does not know that some of its neighbors departed from the overlay, it might forward packets to non-existing neighbors, which leads to a loss of forwarding capacity. We do not explicitly consider this artifact, as it can be easily modelled as the decrease of the forwarding rate of the peers. Second, a peer that arrives before the beginning of time slot i starts playback with packet i at time slot $i + B$, consequently the peer does not request packets with a sequence number lower than i. Third, the number of neighbors a peer has is not constant, but varies over time as nodes join and leave. In the following we show how to estimate the effects of the change of the number of neighbors on the overlay's performance.

For simplicity we consider that nodes arrive to the overlay according to a Poisson process with intensity λ and their lifetime follows an exponential distribution with mean $1/\mu$. Each peer joining the overlay is assigned d neighbors uniformly at random from the nodes that have less than d neighbors. Let us denote by the r.v. $D(t)$ the number of neighbors of a peer at time t ($D(t) \in \{0, \dots, d\}$, $t \in [0, \infty)$) and by the r.v. D_i the number of neighbors of a peer at the beginning of time slot i ($D_i \in \{0, \dots, d\}$). In the following we derive the

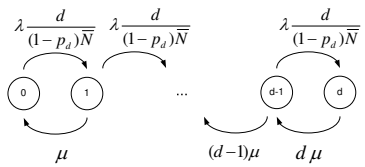

Fig. 3 Markov process used to model the evolution of the number of neighbors of a peer.

distribution of D_i in the steady state of the system, which is the same as the steady state distribution of $D(t)$, i.e., the probabilities $p_z = \lim_{t\to\infty} P(D(t) = z)$, $z \in \{0, \ldots, d\}$.

The average number of peers in the overlay in steady state is $\overline{N} = \lambda/\mu$ and we can approximate the evolution of $D(t)$ with a one dimensional continuous time Markov process as shown in Figure 3. We can approximate the arrival rate of the neighbors to a peer already in the overlay by $\lambda \cdot d/(\overline{N}(1-p_d))$, i.e., in the denominator we use the average of the number of peers with less than d neighbors instead of its actual value. We use an iterative method to calculate the steady state distribution of the Markov process: we start with an initial value of $p_d = 0$ to derive the next value of the steady state probability of p_d until the value converges.

The evolution of P_i^j for an arbitrary peer r depends on the number of neighbors of the node itself and on π_i^j, which in turn depends on the number of neighbors of the neighboring peers of node r. The exact calculation of P_i^j has a complexity of d^d and is computationally not feasible. Hence we make three approximations. First, we approximate the number of neighbors of all the neighboring peers of r with $E[D_i]$ when calculating π_i^j. Second, we assume that the content of the playout buffer of the arriving nodes is statistically identical to that of the nodes already present in the overlay, i.e., can be described by P_i^j. Third, to calculate the probability that the peer receives a packet directly from the root we use $\overline{N}$ instead of the actual overlay size in timeslot i. We evaluate the effect of these approximations in Section 5 by simulation.

The distribution of the number of neighbors of peer r in an arbitrary slot is given by the steady state distribution of D_i, hence for an arbitrary peer in a dynamic overlay instead of (2) we can write

$$P_i^j = \begin{cases} 0 & i < j \\ m/\overline{N} & i = j \\ \sum_{z=0}^{d} P(D_i = z) \cdot \{P_{i-1}^j + & \\ \quad (1 - P_{i-1}^j) \cdot (1 - \pi_i^j)^z\} & i > j \end{cases} \quad (17)$$

5 Performance Evaluation

In the following we first validate our modelling assumptions via simulations, then we evaluate the performance of the four forwarding schemes based on analytical and simulation results.

5.1 Simulation methodology

For the simulations we developed a packet level event-driven simulator. We construct the overlay as follows. Each peer wishing to join sends a connection request message to the boot-strap node, which may be the source node. The boot-strap node responds with a list of randomly selected peers that are already part of the overlay. The joining peer contacts the peers in this list in order to establish neighbor relationships with at least d_{min} but not more than d of them, where $d_{min} = 0.85d$. If it cannot find d_{min} neighbors, the peer issues a new request to the boot-strap node. If, after the second request, the peer still cannot connect to d_{min} neighbors, it issues a force connection request to random peers in the overlay; those peers are forced to drop one of their neighbors and replace it with the joining peer.

For the simulations we first construct an overlay with N peers, thereafter the data distribution starts. For simulations with node churn, nodes join the overlay according to a Poisson process, and the life time distribution is exponential. We run the simulations for 2000 time slots, and the simulation results shown are the averages of 10 simulation runs.

5.2 Model validation

As a first step we validate our assumption that buffer maps are statistically identical, which means that all peers have the same probability of possessing any particular packet at any time. Instead of evaluating the entire buffer map, we compare the probabilities that a packet is played out successfully, that is, it is received by the playout time.

In Figure 4 we show the histogram of the probability of successfully played out packets across the peers, considering the four forwarding schemes and the case of perfect information ($u = 1$). The vertical line shows the mean value of the playout probability averaged over all peers, while in the legend we show its standard deviation. When the number of neighbors of a peer is very low, such as $d = 2$, the probabilities are dispersed over a wide range, as can be seen both visually as well as by the standard deviation. It means that our assumption on statistically identical buffer maps does not hold. As d

increases, the probabilities for different peers approach the mean. This shows that for reasonably high values of d our modelling assumption holds and the results obtained by the model should be accurate. Figure 5 shows the histogram of the playout probability for the four algorithms and for the case where nodes have outdated information about their neighbors' playout buffers with parameters $u = 4$ and $d = 50$. Compared to the perfect information case the standard deviation is higher for all schemes. We see significant effect of delayed buffer maps in the case of fresh-data-first schemes, where the initial statistical difference among the playout buffers is amplified due to the rather deterministic forwarding decisions.

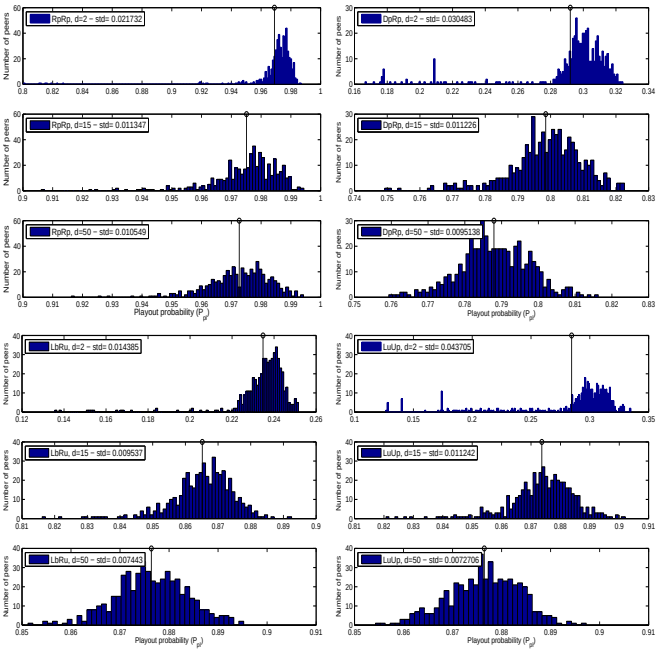

Fig. 4 Histogram of the playout probability for the considered forwarding schemes for an overlay of $N = 500$, $m = 11$, $B = 40$ and for $d = 2, 15$ and 50 respectively.

To investigate the impact of the neighborhood size, d, on the overlay performance, we show in Figure 6 the probability of successful play out versus the number of neighbors d for all considered schemes. The figure contains both analytical and simulation results. For small values of d there is a discrepancy between the model and the simulations, since in this region the assumption on statistically identical buffer maps does not hold. Nevertheless, as d increases the analytical results approach the simulation results. For the case of perfect information, $u = 1$, and for values of $d > 10$ the approximation is very good for all schemes, whereas for $u > 1$ the analytical and the simulations results converge at a higher value of d. We experience slower convergence in the case

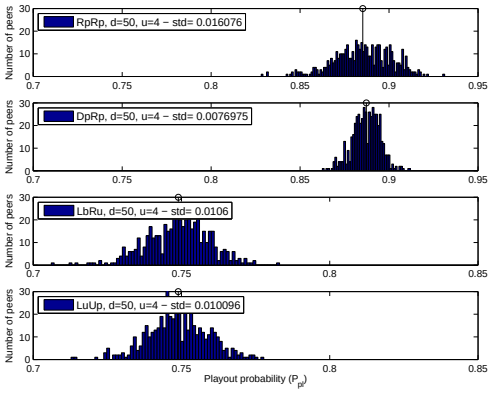

Fig. 5 Histogram of the playout probability in the presence of delayed buffer map updates. Overlay of $N = 500$, $m = 11$, $B = 40$ and $u = 4$.

of fresh-data-first schemes as expected from the standard deviation values on Figure 5.

Based on the figures we can also draw the important conclusion that the playout probability is insensitive to the increase of the neighborhood size above a certain value of d. This suggests that a peer can reach the best achievable performance in terms of playout probability by maintaining a reasonable number of neighbors, and a slight variation of d caused by churn would not lead to variations in the performance.

5.3 Playout probability and playback delay

Next, we investigate the performance of the four forwarding schemes in terms of the ratio of successfully played out packets for static overlays. We consider an overlay of size $N = 500$, and the number of neighbors is $d = 50$. In Figure 7 we show the ratio of successfully played out packets versus the playback delay in the case of perfect information. In all the cases, we can see that the analytical results are very close to the simulation results. For the fresh-data-first and the *DpRp* schemes there is a perfect match, while for *RpRp* the model slightly underestimates the playout probability for small playback delays.

For small buffer lengths the fresh-data-first schemes perform considerably better than the random-packet schemes, which is in accordance with the results presented in [18,21] and confirms the low diffusion delays achieved by the fresh-data-first schemes. The ratio of successfully played out packets remains however low for *LbRu* and *LuUp* even at high playback delays. The reason of this limited performance is the packet selection scheme: since only fresh packets are considered for transmission, the probability that more than one neighbor of a peer transmits the same packet in the same

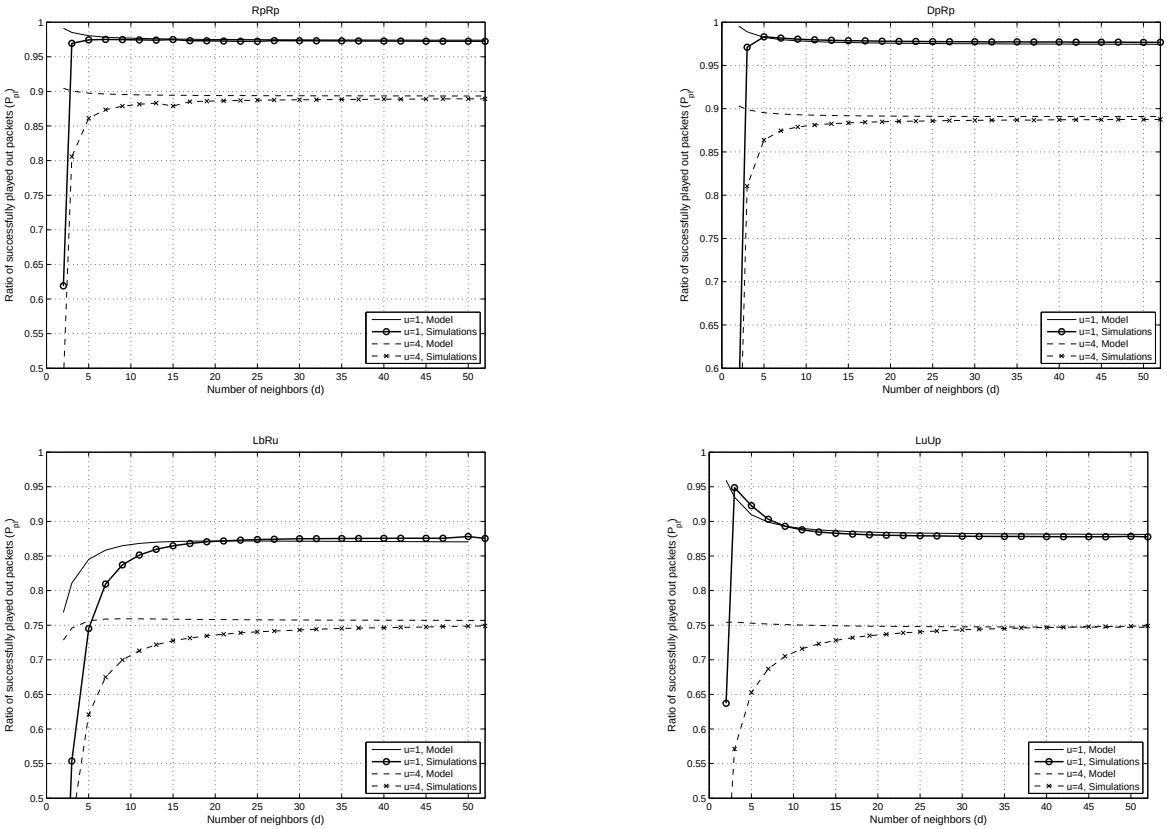

Fig. 6 Probability of successfully played out packets vs number of neighbors. Overlay with $N = 500$, $m = 11$, $B = 40$. Model and simulations.

time slot is relatively high and is not affected by the size of the playout buffer. This result does not contradict the results of [18], since there the same schemes are considered assuming that scheduling decisions are shared without delay. As collision is avoided with this assumption, the playout probability reaches one.

The playout probability increases with the playback delay for the *RpRp* and the *DpRp* schemes, and converges to one. In these cases the randomness of packet selection increases with B and consequently the probability of peers sending the same packet to the same peer in the same time slot approaches zero.

The *LbRu* and *LuUp* schemes yield almost the same performance for all buffer length values, except for large values of B, when the performance of *LuUp* is marginally better. Similarly, *DpRp* slightly outperforms *RpRp* for large values of B as it makes maximum use of the increase of the set of eligible packets.

Next we investigate why data diffusion is slower in the random-packet schemes. Figure 8 shows the cumulative distribution function of the number of overlay hops from the source node for a specific packet. The results shown here are obtained by tracking one randomly selected packet in 10 simulation runs over 10 different overlays. We observe that using the fresh-data-

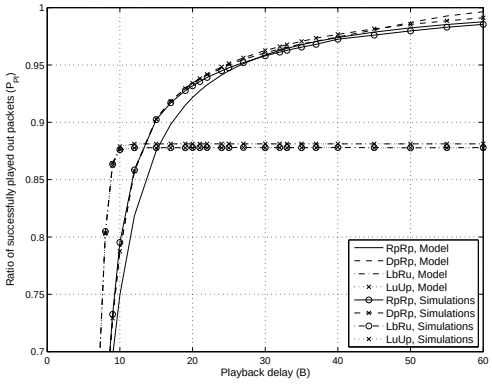

Fig. 7 Ratio of successfully played out packets vs playback delay for four forwarding schemes. Overlay with $N = 500$, $m = 11$, $d = 50$ and $u = 1$. Model and simulations.

first schemes peers receive packets after somewhat fewer hops, which indicates that the trees that packets traverse to reach the peers have slightly higher fanout. This small difference alone does not explain the better delay performance of fresh-data-first schemes. The main reason is that in these schemes the time between receiving a packet and forwarding it to a neighbor is shorter as well. Nevertheless, the random-packet schemes might

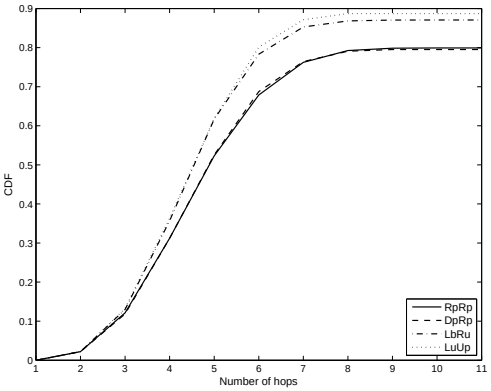

Fig. 8 CDF of the number of hops from the source. Overlay with $N = 500$, $m = 11$, $d = 50$, $B = 10$ and $u = 1$. Simulations.

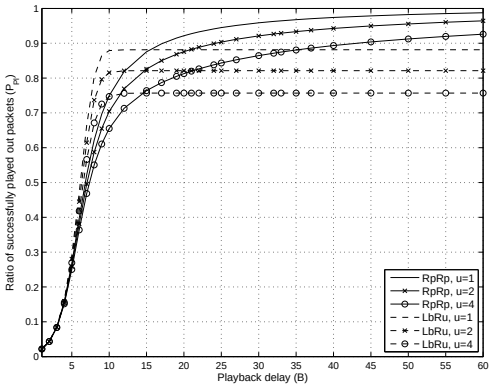

Fig. 9 Ratio of successfully played out packets vs playback delay in the presence of outdated buffer maps. Overlay with $N = 500$, $m = 11$, $d = 50$. Model.

disseminate the data at a lower pace, but at relaxed playout delay constraints it reaches a larger portion of the peers in the overlay.

5.4 Playout probability and outdated buffer maps

Intuitively, the outdated information contained in a buffer map leads to sub-optimal forwarding decisions, and hence to the decreased efficiency of the forwarding algorithms. We show results that confirm this intuition in Figure 9 for *RpRp* and *LbRu*. We observe that the curves that correspond to the case of outdated information ($u > 1$) are similar in shape to the case with perfect information ($u = 1$), but the playout probability is always lower. For small playback delays the impact of outdated information seems to be rather small, however at larger delays the difference becomes significant. Figure 10 shows how fast the playout probability converges to the lower bound, when scheduling decisions are made without buffer map information (that is when $u \to \infty$ in the analytical model). The playout probability decreases fast with the increase of u, which indicates that the efficiency of the forwarding algorithms is very sensitive to the timeliness of the buffer content information. The faster decrease in the case of *LbRu* is again due to the fresh-data-first nature of the scheme. Between buffer map updates information about older packets is still available in the buffer, but those old packets are not considered for transmission.

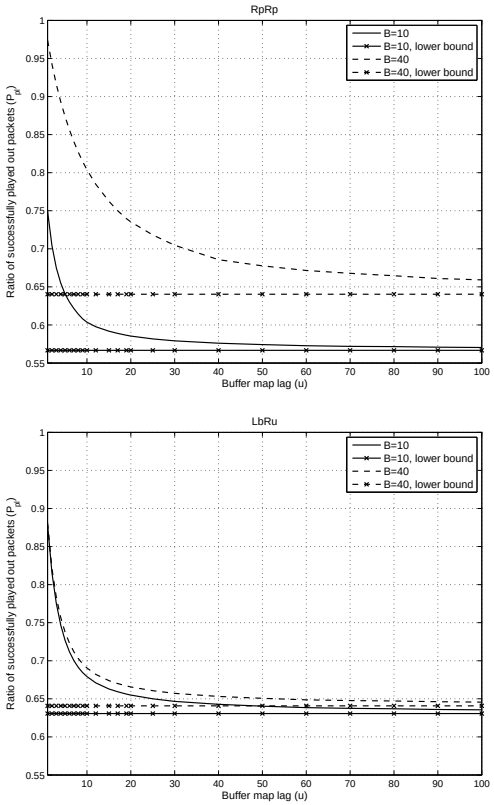

Fig. 10 Ratio of successfully played out packets vs buffer map update lag. Overlay with $N = 500$, $m = 11$, $d = 50$. *RpRp* and *LbRu* schemes. Model.

5.5 Scalability

In this subsection we evaluate the scalability of the considered schemes in terms of the overlay's size. Figure 11 shows the minimum playback delay required to achieve a playout probability of 0.85 as a function of the overlay size N for the *RpRp* and the *LbRu* schemes. The *DpRp* and the *LuUp* schemes give similar results. The horizontal axis is in logarithmic scale. We see that the increase of the necessary playback delay is proportional to $log(N)$ for both schemes and even for $u > 1$ if the considered playout probability is achievable. In [19] the authors showed that the increase of the playback de-

lay required to maintain the playout probability unchanged is proportional to the increase of the depth of the overlay. Hence, we can conclude that all the considered push-based scheduling schemes lead to data distribution trees with a depth that is logarithmic in the number of peers in the overlay.

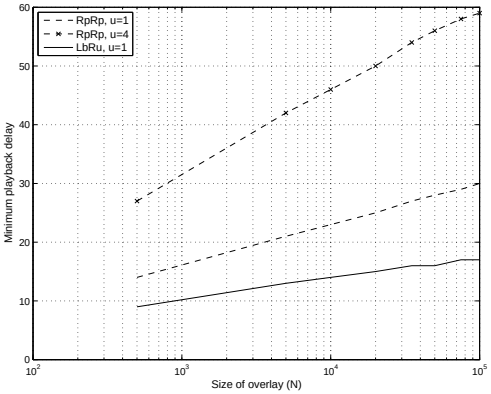

Fig. 11 Minimum playback delay required to achieve a playout probability of 0.85 versus overlay size. *RpRp* and *LbRu* schemes. Model.

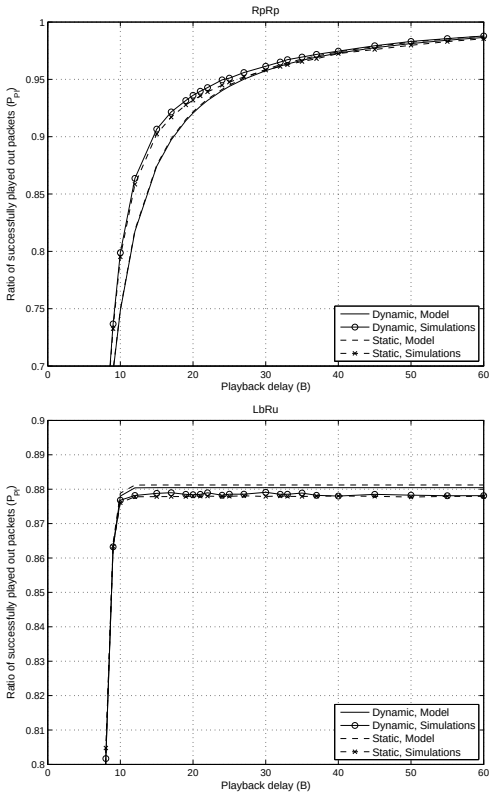

Fig. 12 Ratio of successfully played out packets vs playback delay under churn. Overlay with N=500, m=11 and d=50, $\frac{1}{\mu}$=50 slot. Model and Simulations.

5.6 Effects of node churn

In this subsection we evaluate the effects of node churn on the system performance. Figure 12 shows the playout probability as a function of the playback delay for the *RpRp* and the *LbRu* schemes with and without node churn. The figure shows that the playout probability is only slightly affected by node churn. This result is in accordance with Figure 6: node churn does not decrease the performance of the overlay as long as peers are able to maintain a certain number of neighbors. A consequence of the insensitivity to the evolution of the number of neighbors is that the assumption on the node life time distribution does not affect our results (the steady state probability distribution of the M/G/∞ queue is known to be insensitive to the service time distribution).

Surprisingly however, the simulation results show that the overlay may perform slightly better in the presence of node churn. Clearly, the improved performance cannot be a consequence of the variation of the number of neighbors of the nodes. In order to understand the reason for the improved performance we ran simulations with *altruistic* peers. An *altruistic* peer behaves slightly different than the *conservative* peers considered until now. An altruistic peer that arrives before the beginning of slot i requests packets with a sequence number at least $i - B/2$ from its neighbors in order to help the data distribution, even though itself starts playout only at slot $i + B$ with packet i.

The results of the simulations performed with the *altruistic* peers and the *conservative* peers can be seen in Figure 13, which shows the playout probability as a function of the mean lifetime of peers measured in time-slots. The static case is shown as reference. The lower the mean lifetime, the higher the churn rate, i.e., the more dynamic is the scenario. Contrary to what one would expect, the overlay may benefit from high churn rates if peers are *conservative*, but high churn leads to decreased performance if peers are *altruistic*. We explain this phenomenon by considering an arbitrary neighbor s of a *conservative* peer that joins the overlay at slot i. The peer is interested in packets that are generated at or after slot i. Consequently, the neighboring nodes of s that were already in the overlay have fewer contestants for the packets that were generated before slot i, and may get those packets sooner.

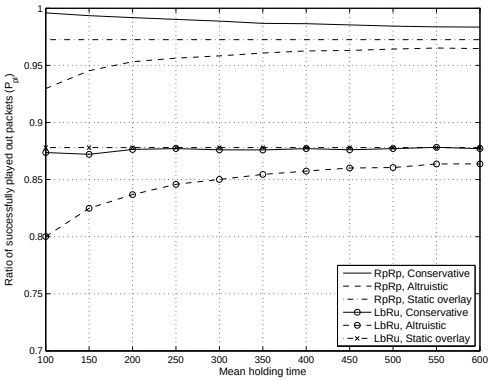

Fig. 13 Ratio of successfully played out packets versus mean holding time for the *RpRp* and *LbRu* schemes. Simulations.

6 Conclusion

In this paper we proposed an analytic framework to assess the performance of various scheduling algorithms for mesh-based P2P streaming systems. The modelling framework is based on the observation that with peers maintaining a fair number of neighbors, the local scheduling decisions may generate per packet distribution trees that are very different and as a result the position of the peers in the trees is statistically the same. Using this framework, we derived analytic performance measures for four forwarding schemes. We proved that fresh-data-first schemes, though able to diffuse data fast, have a limit on the ratio of peers that they can deliver data to. In contrast, the random-packet schemes can achieve a good playout probability and playback delay trade-off. We also evaluated the performance of the forwarding algorithms under outdated information and showed that outdated information quickly leads to a significant decrease of the system's performance. Furthermore, we developed a model that shows that in a dynamic overlay the variation of the number of the peers' neighbors does not affect the overlay performance as long as peers are able to maintain a fair number of neighbors. We also showed that in a dynamic environment, newly arriving nodes can actually act in a beneficial way for the system, by properly adjusting their initial interest window: node arrivals increase slightly the average playout probability compared to a static overlay with the same characteristics. The analytic framework of this paper can be used to evaluate various forwarding solutions and their combinations, and also the effect of network parameters. In our future work we will extend the framework to propose and evaluate scheduling solutions for overlays with heterogeneous uplink and downlink capacities.

References

1. PPLive, http://www.pplive.com/en/about.html.
2. Octoshape, http://www.octoshape.com.
3. UUSee, http://www.uusee.com.
4. X. Hei, Y. Liu, and K. Ross. IPTV over p2p streaming networks: The mesh-pull approach. *IEEE Communications Magazine*, 46(2):86–92, February 2008.
5. V. Fodor and Gy. Dán. Resilience in live peer-to-peer streaming. *IEEE Communications Magazine*, 45(6):116–123, June 2007.
6. S. Birrer and F. Bustamante. A comparison of resilient overlay multicast approaches. *IEEE Journal on Selected Areas in Communications*, 25:1695–1705, December 2007.
7. B. Li, S. Xie, G.Y. Keung, J. Liu, I. Stoica, H. Zhang, and X. Zhang. An empirical study of the coolstreaming+ system. *IEEE Journal on Selected Areas in Communications*, 25:1627–1639, December 2007.
8. X. Hei, C. Liang, Y. Liu, and K.W. Ross. A measurement study of a large-scale p2p IPTV system. *IEEE Transactions on Multimedia*, 9:1672–1687, December 2007.
9. R. Rejaie N. Magharei. Prime: Peer-to-peer receiver-driven mesh-based streaming. In *Proc. of IEEE INFOCOM*, 2007.
10. X. Zhang, J. Liu, B. Li, and T-S. P. Yum. Coolstreaming/donet: a data driven overlay network for efficient live media streaming. In *Proc. of IEEE INFOCOM*, 2005.
11. K. Nahrstedt J. Liang. Dagstream: Locality aware and failure resilient peer-to-peer streaming. In *Multimedia Computing and Networking (MMCN)*, 2006.
12. Y. Tang M. Zhang, L. Zhao, J-G. Luo, and S-Q. Yang. Large-scale live media streaming over peer-to-peer networks through the global internet. In *Proc. ACM Workshop on Advances in peer-to-peer multimedia streaming (P2PMMS)*, 2005.
13. L. Bracciale, F. Lo Piccolo, D. Luzzi, S. Salsano, G. Bianchi, and N. Blefari-Melazzi. A push-based scheduling algorithm for large scale p2p live streaming. In *Proc. of QoS-IP*, 2008.
14. M. Faloutsos A. Vlavianos, M. Iliofotou. Bitos: Enhancing BitTorrent for supporting streaming applications. In *Proc. of IEEE INFOCOM*, 2006.
15. Y. Liu. On the minimum delay peer-to-peer streaming: how realtime can it be? In *Proc. of MM'07*, 2007.
16. S. Tewari and L. Kleinrock. Analytical model for bittorrent-based live video streaming. In *Proc. of IEEE NIME 2007 Workshop*, 2007.
17. L. Massoulie et al. Randomized decentralized broadcasting algorithms. In *Proc. of IEEE INFOCOM*, 2007.
18. T. Bonald, L. Massoulié, F. Mathieu, D. Perino, and A. Twigg. Epidemic live streaming: Optimal performance trade-offs. In *Proc. of ACM SIGMETRICS*, 2008.
19. Gy. Dán and V. Fodor Delay Asymptotics and Scalability for Peer-to-peer Live Streaming In IEEE Trans. on Parallel and Distributed Systems,to appear
20. V. Fodor and I. Chatzidrossos. Playback delay in mesh-based peer-to-peer systems with random packet forwarding. In *Proc. of 1st International Workshop on Future Multimedia Networking (FMN)*, 2008.
21. C. Liang, Y. Guo, and Y. Liu. Is random scheduling sufficient in p2p video streaming? In *Proc. of the 28th International Conference on Distributed Computing Systems (ICDCS)*, 2008.