Modeling Loss and Delay in Multi-tree-based Overlay Multicast

György Dán and Viktória Fodor
Laboratory for Communication Networks
School of Electrical Engineering
KTH, Royal Institute of Technology
Stockholm, Sweden
E-mail: {gyuri,vfodor}@ee.kth.se

Abstract

In this paper we propose an analytical model to evaluate the end-to-end loss and delay characteristics for live multicast streaming. We consider multiple-tree architectures and push based transmission. We give an asymptotic bound on the performance of large overlays, and present a necessary condition for the overlay to be stable. Based on the model we show how the overlay's loss characteristics is influenced by the number of distribution trees, the error control solutions used, the allocation of the bandwidth resources of the peers between the trees and the number of nodes in the overlay. Based on our results, we propose a tree structure that improves the stability of the overlay. We use the model to show how the structure of the overlay and the available bandwidth influence the delay characteristics of the data distribution and the playout buffer requirements.

Index terms: Modeling, Push-model, Data distribution performance, Forward error correction

[0]Technical Report TRITA-EE 2007:016, March 2007

School of Electrical Engineering, KTH, Royal Institute of Technology

1 Introduction

The success of peer-to-peer overlays for live multicast streaming will depend on their ability to maintain acceptable perceived quality at all the peers, that is, to provide data transmission with low delay and information loss. Since peers have to relay data to their children with low delay, the possibilities of error recovery are limited. Consequently, the main problem to be dealt with is the propagation and thus the accumulation of losses, which results in low perceived quality for peers far from the source.

The architectures proposed for peer-to-peer streaming generally fall into one of two categories: push based or pull based [1]. Solutions in both categories utilize multi-path transmission to ensure graceful quality degradation in dynamic overlays, where peers can leave during the streaming session. Multi-path transmission offers two advantages. First, disturbances on an overlay path lead to graceful quality degradation in the nodes. Second, the output bandwidth of the peers can be utilized more efficiently.

Several works deal with the management of such overlays: giving incentives for collaboration, peer selection, and tree reconstruction considering peer heterogeneity and the underlying network topology ([2, 3, 4, 5, 6, 7] and references therein). There are also numerous proposals on how to improve the robustness of the overlays to errors: In [8] the authors propose time shifting and video patching to deal with losses and discuss related channel allocation and group management issues. In [9] robustness is achieved by distributing packets to randomly chosen neighbors outside the distribution tree. In [10] retransmission of the lost data is proposed to limit temporal error propagation. CoopNet [11] and SplitStream [12] propose the use of multiple distribution trees and a form of multiple description coding (MDC) based on forward error correction (FEC). In the case of packet losses peers can stop error propagation by reconstructing the video stream from the set of received substreams using error correcting codes. Several proposed designs were also implemented (e.g.,[13] and [2] and references therein), but the evaluation of these solutions is mostly based on simulations and small scale measurements.

We focus on solutions that apply the push model for data distribution and FEC to deal with packet losses due to congestion and peer departures. Our goal is to define abstract models of peer-to-peer streaming overlays in order to give an understanding of the basic characteristics of streaming over multiple transmission trees and thus, support future system design. We consider a generalized multiple-tree-based overlay structure that allows peers to contribute in a given number of trees [14]. Special cases of this architecture are the ones proposed in [3, 6, 11, 12, 15]. and analyzed in [16, 17].

The results presented in this paper are twofold. First, we show that FEC has limited capabilities in large overlays and propose an overlay structure where *the spread between the layers* where a node participates in the different trees *is limited* in order to improve the effectiveness of FEC. Second, we give an abstract model of end-to-end delay and identify its primary sources in the considered multiple-tree-based systems.

The rest of the paper is organized as follows. Section 2 describes the considered overlay structure and error correction scheme. We evaluate the stability of the data distribution in Section 3. Section 4 discusses the performance of the overlay based on the mathematical models and simulations and we conclude our work in Section 5.

2 System description

The overlay consists of a root node and N peer nodes. The peer nodes are organized in t distribution trees. Each peer node is member of at least one tree, and in each tree it has a different parent node from which it receives data. We say that a node that is l hops away from the root node in tree e is in layer l of tree e. We denote the maximum number of children of the root node in each tree by m, and we call it the multiplicity of the root node.

Nodes can have children in up to d of the t trees, called the fertile trees of a node. A node is sterile in all other trees, where it does not have any children. (d is a system parameter.) We call the number of cogs of the node the number of children that the node is willing to forward data to. If a node r has enough capacity to forward data to γ^r children then we say that the node has a total of γ^r cogs in its fertile trees and has no cogs in its sterile trees. For $d > 1$ the nodes balance their cogs between trees, i.e., a node can have up to $\lceil \gamma^r/d \rceil$ cogs in each of its fertile trees. We call an overlay well-maintained if the number of fertile nodes is maximal in every layer of its trees. Well-maintained overlays have the smallest depth for given N, t and d. For instance, in a well-maintained overlay with L layers, each node is $1 \leq l \leq L$ hops away from the root node in its fertile trees, and $L-1 \leq l \leq L$ hops away in its sterile trees.

We define the *layer spread*, δ^r, as the difference between the uppermost layer where node r is fertile and the lowermost layer where it is sterile. Without any restrictions $\delta^r \leq L-1$. We define an overlay structure with *layer spread limit* δ as one where for each node $\delta^r \leq \delta$. Limiting the layer spread can increase the number of layers in the overlay, but since the fraction of the sterile nodes in a layer is proportional to $(t/d)^{-\delta+1}$, the increase is low for reasonable values of t.

By setting $d = t$ one gets the minimum breadth trees described in [11], and by setting $d = 1$ one gets the minimum depth trees evaluated in [3, 6, 11, 12, 18]. For $1 < d < t$ the number of layers in the overlay is $O(logN)$ as for $d = 1$. Fig. 1 and 2 show an overlay for $t = 2$, $m = 2$ and $d = 1$ without and with limited layer spread respectively.

2.1 Tree management

The construction and the maintenance of the trees can be done either by a distributed protocol (structured, like in [12] or unstructured, like in [10]) or by a central entity, like in [11]. The purpose of the tree maintenance algorithm is to find eligible parents for the nodes (arriving nodes, preempted nodes and nodes disconnected due to the departure of a parent) based on the parent selection criteria, such as closeness to the root and the priorities of the nodes. The results presented in this paper do not depend on the particular algorithm used, our focus is on the performance of the overlay as a function of its structure, rather than the efficiency of the tree maintenance algorithm. In Section 4 we briefly describe the tree maintenance algorithm used for the simulations .

2.2 Data transmission and error resilience

The root splits the data stream into t stripes, with every t^{th} packet belonging to the same stripe, and it sends the packets at round-robin to its children in the different trees. Peer nodes relay the packets upon reception to their respective child nodes. The root uses block based FEC, e.g., Reed-Solomon codes [19], so that nodes can recover from packet losses due to network congestion and node departures. To every k packets of information c packets of redundant information are added resulting in a block length of $n = k + c$. We denote this FEC scheme by FEC(n,k). Lost packets can be reconstructed as long as no more than c packets are lost out of n packets. Once a node receives at least k packets of a block of n packets, it may recover the remaining c packets. If a packet, which should have been received in the tree where the node is fertile, is recovered, then it is sent to the respective children. Duplicate packets are discarded by the nodes. If the root would like to increase the ratio of redundancy while maintaining its bitrate unchanged, then it has to decrease the source rate. If $n \leq t$ then at most one packet of a block is distributed over the same distribution tree. Using this FEC scheme one can implement UXP, PET, or the multiple description coding (MDC) scheme considered in [11].

3 Data distribution model

The metric we use to measure the performance of the data distribution is the probability π that an arbitrary node receives or can reconstruct (i.e., possesses) an arbitrary packet. If we denote by ρ_r the number of packets possessed by node r in an arbitrary block of packets, then π can be expressed as the average ratio of packets possessed in a block over all nodes, i.e., $\pi = E[\sum_r \rho_r/n/N]$.

The mathematical model we present describes the behavior of the overlay in the presence of correlated packet losses and without node dynamics. We model the losses on the input links of the nodes and denote the probability that a packets in a block of j packets are lost by $P(a, j)$. For example, $P(a, j)$ can be calculated using a two-state Markovian model, often referred to as the Gilbert model, given the packet loss probability p and the conditional loss probability (the probability that a packet is lost given that the previous packet was lost) $p_{\omega|\omega}$ as shown in [16]. We assume that the probability that a node is in possession of a packet is independent of whether a node in the same layer is in possession of a packet. For simplicity, we assume that $n = t$, and that $\gamma^r \geq t$ is equal for all nodes. Let us denote by L the number of layers in the overlay. We assume that nodes are in the same layer in their fertile trees, and in their sterile trees respectively, and we introduce $L_l = min(L, l + \delta)$ the layer where a node that is fertile in layer l is located in its sterile trees. We will comment on the possible effects of our assumptions and on the possible extensions of the model in Section 3.3.

The probability that nodes receive data from other nodes is determined by the probability that a node that forwards data in a tree can forward the data to its children. Hence, we introduce the probabilities $\pi_f(l)$ that a node that is in layer l in its fertile tree, receives or can reconstruct an *arbitrary packet in its fertile tree*. The probability that a node in layer l receives a packet in a tree is $\pi_a(l) = P(0,1)\pi_f(l-1)$. A node can possess a packet in its fertile tree either if it receives the packet or if it can reconstruct it using the packets received in the other trees. Reconstruction can take place if the number of received packets is at least k out of the remaining $n-1$, hence we can write for $1 \leq l \leq L$

$$\pi_f(l) \;=\; \pi_a(l) + (1-\pi_a(l)) \sum_{j=k}^{n-1} \sum_{u=max(0,j-n+d)}^{min(j,d-1)} R_f(l,d-1,u) R_s(L_l, n-d, j-u), \tag{1}$$

where the probabilities

$$\begin{aligned} R_f(l,d-1,u) &= \binom{d-1}{u} \pi_a(l)^u (1-\pi_a(l))^{d-1-u}, \\ R_s(l,n-d,j-u) &= \sum_{h=j-u}^{n-d} \pi_f(l-1)^h (1-\pi_f(l-1))^{n-d-h} P(n-d-j+u, n-d), \end{aligned}$$

are the probabilities that the node receives packets in u of its $d-1$ fertile trees and that it receives packets in $j-u$ of its $n-d$ sterile trees respectively. The model considers loss correlations in the sterile trees only through R_s, but as we will see later, even this approximation overestimates the effects of correlations. Based on the probabilities $\pi_f(l)$ we can express $\pi(l)$ $(1 \leq l \leq L-1)$, the probability that a node that is fertile in layer l possesses an arbitrary packet. If a node receives at least k packets in a block of n packets then it can use FEC to reconstruct the lost packets, and will hence possess all n packets. Otherwise, FEC cannot be used to reconstruct the lost packets. Packets can be received in the d fertile trees and in the $t-d$

sterile trees. Hence for $\pi(l)$ we get the equation

$$\pi(l) = \frac{1}{n}\sum_{j=1}^{n-d}\sum_{u=1}^{d}\tau(j+u)\binom{d}{u}R_f(l,d,u)R_s(L_l,n-d,j), \tag{2}$$

where $\tau(j+u)$ indicates the number of packets possessed after FEC reconstruction if $j+u$ packets have been received:

$$\tau(j+u) = \begin{cases} j+u & 0 \leq j+u < k \\ n & k \leq j+u \leq n. \end{cases}$$

We use an iterative method to calculate the probabilities $\pi_f(l)$. Since the root node possesses every packet, we have that $\pi_f(0) = 1$. We initialize $\pi_f(l)^{(0)} = 0$ for $1 \leq l \leq L$. Then, in iteration *i*, we calculate $\pi_f(l)^{(i)}$, $1 \leq l < L$ using the $\pi_f(l)^{(i-1)}$. The iteration can stop in two cases: when $\pi_f(L-1)$ approaches its limit, i.e., $|\pi_f(L-1)^{(i-1)} - \pi_f(L-1)^{(i)}| < \varepsilon$, where $\varepsilon > 0$ and $i > L$, or after reaching a certain number of iterations. We will use the second stop criterion to investigate the temporal evolution of π in Section 4.3. The $\pi(l)$ can then be calculated using (2).

We can express π as the sum of the $\pi(l)$ weighted by the number of nodes that are fertile in the respective layers N_l

$$\pi = \frac{1}{N}\sum_{l=1}^{L} N_l \pi(l). \tag{3}$$

3.1 Modeling the temporal evolution of $\pi_f(l)$

There are two sources of delay in the considered overlay. First, the FEC decoding delay. For a source with average bitrate *B*, average packet size *b* and block length *n* the mean block delay is $E[D_b] = nb/B$.

Second, the time it takes for a packet to travel one hop in the overlay, the per-hop-delay D_{ph}. The per-hop-delay itself consists of four components. First, the queuing and propagation delays between the nodes, denoted by D_p, not including the queuing delays on the access links. The tree maintenance algorithm can aim at minimizing this delay.

Second, the queuing delays on the input links of the nodes. The mean queuing delay on the input link of a node with input bandwidth C_{in} can be expressed as

$$E[D_{tr,i}] = \overline{W}_{in} + b/C_{in} \geq b/C_{in}, \tag{4}$$

where $\overline{W}_{in}$ is the mean waiting time of the packets in the input link's buffer. The input link's buffer can be modeled by a G/D/1 queue (assuming constant packet sizes), and the delay can be negligible if the nodes' input capacities are much higher than the stream's bandwidth.

Third, the queuing delays on the output links of the nodes. We can model the output queue as a $\mathrm{GI}^{\mathrm{X}}/\mathrm{D}/1$ queue with batch arrivals of constant size γ^r/d [20]. The mean queuing delay on the output link of a node with upload bandwidth C_{out} is

$$E[D_{tr,o}] = \overline{W}_{out} + b\gamma^r/d/2/C_{out} \geq b\gamma^r/d/2/C_{out}, \tag{5}$$

where $\overline{W}_{out}$ is the mean waiting time of the first packets of the arriving batches of packets in the output buffer of the node. If we denote by u the link's utilization, i.e., $\gamma^r = uC_{out}/Bt$, then we have

$$E[D_{tr,o}] = \overline{W}_{out} + ubC_{out}/Bt/d/2/C_{out} = \overline{W}_{out} + ubt/d/2/B. \tag{6}$$

In overlays where the number of cogs of the nodes is proportional to their output bandwidth [3, 6], i.e., u is equal for all nodes, the second term on the right hand side of (6) is *independent of the distribution of the nodes' output bandwidths*. As bandwidth resources are usually scarce in overlay multicast [6, 15], u has to be high to maintain the overlay feasible, and hence $D_{tr,o}$ cannot be neglected.

Fourth, the processing delays in the nodes. For the considered block lengths and common streaming bitrates the processing delays (e.g., arithmetic operations for Reed-Solomon coding) are negligible compared to other sources of delay, and are not considered in this paper.

The successive iterations of the model can be interpreted as the propagation of the packets of a FEC block in the overlay. If we assign the average per-hop delay D_{ph} to each iteration, then the performance predicted by the model after i iterations can be used to approximate the performance of an overlay in which the nodes are equipped with a playout buffer big enough to hold $BE[D(i)]$ data, where

$$E[D(i)] = E[D_b] + iE[D_p + D_{tr,i} + D_{tr,o}]. \tag{7}$$

We validate the accuracy of this approximation in Section 4.3.

3.2 Asymptotic behavior for large N

In the following we give an asymptotic bound on π to better understand its evolution. It is clear that $\pi_f(l)$ is a non-increasing function of l and that $\pi_f(l) \geq 0$. Hence, we can give an upper estimate of $\pi_f(l)$ by assuming that the nodes that forward data in layer l are sterile in the same layer (since $\pi_f(l) \geq \pi_f(L_l)$ as $l \leq L_l$). Then, instead of (1) we get the following nonlinear recurrence equation

$$\begin{aligned} \overline{\pi}_f(l+1) &= \overline{\pi}_a(l+1) + \\ &(1-\overline{\pi}_a(l+1)) \sum_{j=n-c}^{n-1} \sum_{u=max(0,j-n+d)}^{min(j,d-1)} R_f(l,d-1,u) R_s(l,n-d,j-u). \end{aligned} \tag{8}$$

This equation is the same as (2) in [16], and thus the analysis shown there can be applied to describe the evolution of $\overline{\pi}_f(l)$. For brevity, we only state the main results regarding $\lim_{l\to\infty} \overline{\pi}_f(l) = \overline{\pi}_f(\infty)$, for $P(a,j)$ described by the Gilbert model. For a detailed explanation see [16].

For every (n,k) there is a loss probability $p_{max}(p_{\omega|\omega})$ below which the packet possession probability $\overline{\pi}_f(\infty) > 0$ and above which $\overline{\pi}_f(\infty) = 0$. Furthermore, for any $0 < \delta < 1$, p and $p_{\omega|\omega}$ there is (n,k) such that $\overline{\pi}_f(\infty) \geq \delta$.

Consequently, in the considered overlay if $p > p_{max}(p_{\omega|\omega})$, then $\lim_{N\to\infty} \pi = 0$, because $\overline{\pi}_f(l) \geq \pi_f(l) \geq \pi(l)$, and $lim_{N\to\infty} \overline{\pi}_f(L) = 0$. For $p < p_{max}(p_{\omega|\omega})$ stability depends on the number of layers in the overlay and on the FEC block length, because of the initial condition $\pi_f(l)^{(0)} = 0$, $1 \leq l \leq L$, but not directly on the number of nodes. This explains why placing nodes with large outgoing bandwidths close to the root improves the overlay's performance [3, 6].

For an overlay with L layers, a sufficient condition for stability is $(1-p)^L > r_1$, where r_1, $0 < r_1 < 1$ is the unstable fixed point of (8). If the condition is satisfied then π can be bounded from below by $\underline{\pi} = r_2$, where r_2, $0 < r_1 < r_2 < 1$ is the asymptotically stable fixed point of (8). If the sufficient condition is not satisfied then the lower bound is given by not considering FEC reconstruction, i.e. $\underline{\pi} = (1-p)^L$.

3.3 Discussion of the assumptions

In the following we discuss the validity of certain assumptions made in the model. The model does not take into account heterogeneous losses, though it can be extended by following the procedure presented in [16] for the minimum breadth trees. The effects of nodes with heterogeneous input and output bandwidths can be included in the model in a similar way. We decided to show equations for the homogeneous case here to ease understanding.

Our results for block based FEC apply to PET and the MDC scheme considered in [11], where different blocks (layers) of data are protected with different FEC codes. The packet possession probability for the different layers depends on the strength of the FEC codes protecting them, and can be calculated using the model.

The model does not explicitly take into account node departures, an important source of disturbances in overlay multicast. Following the arguments presented in [16] node departures can be incorporated in the model as an increase of the loss probability by $p_d = N_d/N \times \theta$, where N_d is the mean number of departing nodes per time unit and θ is the time nodes need to recover (discovery and reconnection) from the departure of a parent node. The time for recovery can depend on the node's priority in the tree, and hence p_d can be tree dependent. The model can be extended for such a scenario by modifying the functions R_f and R_s. The simulation results presented in [16] suggest that this approximation is accurate for low values of p_d.

The results of the model apply for $n < t$ without modifications, and a similar model can be developed for $n > t$, by modifying eqs. (1-2) for non-binary random variables corresponding to the number of packets received out of the n/t packets distributed in a tree. For $n > t$ node departures appear as consecutive losses in the blocks of packets, and their effect on the performance is more severe than for $n = t$.

4 Performance evaluation

We developed a packet-level event-driven simulator to validate the model. We used the GT-ITM topology generator [21] to generate a transit-stub network with 10^4 nodes and average node degree 6.2. We placed each node of the overlay at random at one of the 10^4 nodes of the topology and used the one-way delays given by the generator between the nodes (mean 67 ms, standard deviation 21 ms, maximum 180 ms). The delay between overlay nodes residing on the same node of the topology was set to 1 ms. The inter-arrival times of nodes are exponentially distributed, this assumption is supported by several measurement studies [22, 23]. The session holding times M follow the log-normal distribution, the mean holding time is $E[M] = 306$ s [22].

Tree maintenance: We assume that a distributed algorithm, such as gossip based algorithms, is used by the nodes to learn about other nodes, and that it provides random knowledge of the overlay such as in [15]. When a node wants to join the overlay, it contacts the root and obtains a random list of $g = 100$ members of every tree. The root tells the arriving node in which trees it should forward data: in the ones with the least amount of forwarding capacity. The arriving node then uses the following parent selection procedure to find a parent.

To select a parent in a tree, the node sorts the g members it is aware of into increasing order according to their distances from the root, and looks for the first node that has available capacity or has a child that can be preempted, i.e., which has lower priority. We limit the layer spread by allowing a node r to preempt any node (even a fertile one) in order to maintain $\delta^r \leq \delta$. Otherwise, fertile nodes can preempt sterile nodes.

If the node has to preempt a child, but itself has available capacity, then the preempted child can immediately become a child of the preempting node. Otherwise, the preempted child has to follow the parent selection procedure just like the child nodes of a departed node.

Data distribution: We consider the streaming of a $B = 112.8$ kbps data stream, and the bandwidths of the nodes' input and output links are set to $C = 128$ kbps, unless otherwise stated. The capacity of the root node's output link is 10 Mbps, and we consider $m = 50$ throughout the paper for easy comparison. As m only influences the overlay's depth, the choice of m does not influence our conclusions. The packet size is 1410 bytes. Unless stated otherwise, the nodes have a playout buffer capable of holding 140 packets, which corresponds to 14 s delay with the given parameters. Every node has an input and an output buffer of 80 packets each to absorb the bursts of incoming and outgoing packets. Apart from packet losses due to the overflow of the input and output buffers and due to late arriving packets, we simulate packet losses on the input links of the nodes using the Gilbert model. For stationary loss probability p and conditional loss probability $p_{\omega|\omega}$ we set the parameters of the model as described in [16].

To obtain the results for a given overlay size $\overline{N}$, we start the simulation with $\overline{N}$ nodes in its steady state as described in [24]. We set $\lambda = \overline{N}/E[M]$ and let nodes join and leave the overlay for 5000 s. The purpose of this warm-up period is to introduce randomness into the tree structure. The measurements are made after the warm-up period during 1000 s and the presented results are the averages of 10 simulation runs. The results have less than 5 percent margin of error at a 95 percent level of confidence.

4.1 Approximating the overlay structure

It is possible to approximate the number of nodes per layer for an arbitrary distribution of γ^r, but since γ^r only influences the number of layers for given N,t,d, we restrict ourselves to the case $\gamma^r = t$ for simplicity. The number of fertile nodes in layer l of a well-maintained tree follows the recurrence $N_l = t/dN_{l-1} - (t-d)N_{l-\delta}$ with initial conditions $N_1 = min(N/(t/d), m)$, $N_l = 0$ for $-\delta < l \leq 0$. For $\delta \geq L-1$ the solution is trivial, $N_l = m(t/d)^{l-1}$. For $\delta < L-1$ a real solution does not always exist for $d > 1$, in which case the overlay is not feasible. The simulated overlay's structure differs from this approximation due to node dynamics, but as our results below show, the difference does not have a significant effect on the accuracy of our model.

4.2 Loss performance

In this section we show results for the limit value of $\pi(l)$ and $\pi(l)$ for $\varepsilon = 10^{-6}$. Fig. 3 shows $\pi(l)$ as a function of l for $t = n = 4$, $c = 1$, $\overline{N} = 10^4$ and independent losses. The value of the threshold for the FEC(4,3) code is $p_{max} = 0.129$. The figure shows that both the upper and the lower bounds are tight when the overlay is stable ($p = 0.10$). However, in the unstable state ($p = 0.14$) $\pi(l)$ drops quickly and the lower bound given is rather loose. Note the effects of the approximation of the number of layers. In the stable state the simulation results match the analytical results, as the number of layers does not influence π. In the unstable state the simulated performance is however slightly worse than the modeled performance due to the higher number of layers in the simulations.

Fig. 4 shows π as a function of p obtained with the mathematical model for $m = 50$, $n = t$, $d = 1$ and independent losses for various block lengths, redundancy rates and overlay sizes. The figure shows that π remains high and is unaffected by N as long as the overlay is stable. It drops however once the overlay becomes unstable, and the drop of the packet possession probability gets worse as the number of nodes and hence the number of layers in the overlay increases. Increasing t (and hence n) increases π in a stable system, but the drop of the packet possession probability gets faster in the unstable state, since longer FEC codes are less efficient at high loss rates. Similarly, increasing the ratio of redundancy increases π and the region of stability, but the drop of π gets faster in the unstable state. The curves corresponding to $\pi(\infty)$ show the value of the asymptotic bound calculated using (8).

By increasing n and t one can increase π, but at the same time the probability that the trees become disconnected (alternatively, the time θ to find a parent) in the case of dynamic overlay membership increases [14]. To avoid the trees to become disconnected, one has to increase d, the number of fertile trees. The depth of the trees does not change if t/d is kept constant. Fig. 5 evaluates this scenario and shows π as a function of p obtained with the mathematical model for $m = 50$, $n = t$ and independent losses for various values of the number of the fertile trees d and number of trees t. The figure shows that by increasing t and d one can improve the overlay's performance in its stable state, but decreases the stability region due to the longer FEC block ($n = t$). We present simulation results to validate our simplifying assumption with respect to the nodes' positions in their fertile trees, and conclude that the model is accurate. We observe slightly worse performance at the transition between the stable and unstable states for $d > 1$, which is due to the finite playout buffer in the simulations.

Next, we evaluate the effect of limiting the layer spread across the transmission trees. Figure 6 shows π as a function of the overlay's size for $k/n = 0.75$ with and without limiting the layer spread. We observe that $t = 4$ has the widest stability region, though it results in a lower value of π in the stable state. Increasing the number of fertile trees decreases the stability region the most, as it combines long FEC blocks with a relatively high number of layers. Limiting the layer spread makes the stability region wider for $t = 16, d = 4$, meanwhile it does not decrease the value of π in the stable state despite the slight increase of the number of layers. Among the considered ones the $t = 16, d = 1$ architecture provides the most shallow trees and thus the widest stability region in terms of number of nodes. There, limiting the layer spread to $\delta = 2$ keeps the overlay stable for the considered range of overlay size. The number of layers corresponding to the transition from the stable to the unstable state (shown with the arrows) is mainly determined by n and k, and increases if the layer spread is limited.

Figure 7 compares π in the case of correlated losses and independent losses (for independent losses $p_{\omega|\omega} = p$). The figure shows that loss correlations slightly decrease the overlay's performance, but not as much as predicted by the model, even though the model only considers the correlation between packets arriving in the sterile trees. Packets of a block do not necessarily arrive back-to-back to a node, hence the correlation between the loss of packets of a block is lower than that between the loss of consecutive packets on the input link of the node (denoted by $P(a, j)$ in the model).

4.3 Delay characteristics

In this section our focus is on how the size of the playout buffer influences the probability of packet possession. We introduce the notion of convergence, and say that the value of π converges after i iterations, if it reaches 99.9 percent of its limit value. First, we use the model to give an understanding of how the number of iterations needed to reach convergence depends on the overlay's parameters and the loss probability. Then we investigate the relationship between the playout buffer's length and the overlay's performance.

4.3.1 Convergence speed

First we consider trees with $N = 10^4$ nodes, for which FEC is effective even without limiting the layer spread according to Fig. 6. Fig. 8 shows π as a function of the number of iterations for different loss probabilities and ratios of redundancy. The minimum number of iterations needed for the data to reach the nodes in their sterile trees in an overlay with L layers is L. For the considered tree parameters $L = 5$, hence we show results for $i \geq 5$. The figure shows that with a higher ratio of redundancy the convergence is much faster for a given loss probability. Similarly, the convergence is faster for lower loss probabilities at a given redundancy rate. In both cases, the reason for the faster convergence is that the overlay is further from the unstable regime. We observe slightly improved performance when limiting the layer spread.

Figure 9 shows the number of iterations needed for convergence as a function of the loss probability for different tree structures, and it supports the conclusion that the number of iterations needed is influenced by both the number of layers, the FEC block length and the distance from the stability threshold.

Figure 10 shows π as a function of the number of iterations for different tree structures. Increasing the number of trees for $d = 1$ decreases the number of layers, and consequently, the number of iterations needed decreases. The convergence speed is slightly improved by the longer FEC blocks as well, as observed for $t = 16, d = 4$ compared to $t = 4, d = 1$ (the number of layers is the same in the two overlays).

To evaluate the convergence in large overlays, Figure 11 shows π as a function of the number of iterations for different overlay sizes. The number of iterations needed to achieve convergence increases with the overlay's size. Hence, if a node observes increasing convergence times but unchanged packet possession probability after convergence, it can infer that the overlay's size is increasing. We see that for $i \approx L$ limiting the layer spread results in inferior performance due to the increased number of layers. This region is however not of practical interest due to the low values of π. For higher values of i and large overlays limiting the layer spread leads to significantly faster convergence as shown for $\overline{N} = 10^5$, and eventually achieves stability as for $\overline{N} = 10^6$.

The overlay's size affects the convergence speed through the number of layers of the overlay, i.e., it is not the number of nodes, similar to the results on stability. Consequently, a tree maintenance algorithm that cannot keep the overlay well-maintained (such as, for example, random selection [18]) can lead to instability already for a relatively small number of nodes. (E.g., the difference in terms of number of layers between $\overline{N} = 10^4$ and $\overline{N} = 10^6$ is only five: for $m = 50$ and $t/d = 4$ we have $L_{10^4} = 5$ and $L_{10^6} = 10$.)

4.3.2 Iteration vs. delay

In this section we evaluate the accuracy of the approximation of the playout buffer's effects on the performance described in Section 3.1. As the playout buffer's length determines the time available for π to converge, we will use playout buffer requirement and convergence time as synonyms in the following. For the analytical results we use the mean propagation delay of the simulations, $E[D_p] = 44$ ms. We show results for $p = 0.10$ and $k/n = 0.75$ for easy comparison. These parameters involve data transmission close to the stability threshold, which in turn means high number of iterations to achieve convergence. The higher the number of iterations in the model, the more irregular the packet arrival process at the peers in the simulations. Hence the considered scenario is a "bad" case in terms of the regularity of the arrival processes.

Deterministic arrivals: First, we consider the model for the best case scenario, when the arrival processes are deterministic both on the input and on the output links of the peers. In this case $\overline{W}_{in} = 0$ and $\overline{W}_{out} = 0$ in (4) and (5) respectively. We consider three scenarios with different utilizations of the input and the output links. In the first scenario ("inf.cap.") the input and output link capacities are $C_{in} = C_{out} = 10$ Mbps (the number of cogs per node is still t), and consequently the per-hop-delay is determined by the propagation delays ($E[D_{ph}] \approx 48\ ms$). In the second scenario ("inf.incap.") the input link capacities are $C_{in} = 10$ Mbps, the output link capacities are $C_{out} = 128$ kbps, i.e., close to the stream's bitrate. The per-hop-delay is increased by the transmission times on the output links ($E[D_{ph}] \approx 221\ ms$). In the third scenario ("fin.cap.") both the input and the output links' capacities are $C_{out} = C_{in} = 128$ kbps, and the per-hop-delay is increased by the transmission times on both links ($E[D_{ph}] \approx 308\ ms$).

Figure 12 shows π as a function of the playout buffer's size for $t = n = 4$, $m = 50$, $p = 0.1$ and $\overline{N} = 10^4$. The time to convergence is to a much lower extent influenced by the propagation delays than by the transmission delays on the output links of the nodes. This conclusion is true for any overlay in which the output link utilizations are high (due to scarce bandwidth resources), *independent of the distribution of the output links' capacities* due to (6). For the considered input bandwidths even the transmission delays on the input links have a bigger impact on the required size of the playout buffer than the propagation delays. We conclude that the achievable gain in terms of decreasing *the convergence time* (i.e., playout buffer requirements) by using a tree-maintenance algorithm that chooses parents based on minimum round-trip-time *depends* very much *on the capacity resources available to the overlay and the output links' utilizations*. Nevertheless, minimizing the round-trip-time can have indirect benefits, e.g., lower loss rates between nearby nodes and better network utilization.

We performed simulations to validate the accuracy of our approximations: the use of the mean propagation and transmission times, the deterministic arrival process and the approximation of the number of layers of the overlay. Fig. 13 shows the results. We observe that the the main characteristics of the analytical and simulation results are the same. For very small playout buffers the model underestimates, for large playout buffer sizes it overestimates the actual performance, due to the variance of the per-hop-delays, which is not considered in the model.

Fig. 14 compares results obtained using the model for different overlay structures. In the case of "infinite" capacities ($C_{in} = C_{out} = 10$ Mbps) the results are similar to those in Fig. 10. Nevertheless, in the case of finite capacities for $t = 16$ the shorter transmission delays on the output links using $d = 4$ result in lower convergence time than using $d = 1$, despite the higher number of iterations needed for convergence. Because of the increased value of $E[D_{tr,o}]$, the convergence time for $t = n = 16$ is higher than that for $t = n = 4$ despite the lower number of layers and less iterations needed for convergence.

We show simulation results for the scenarios with finite capacities in Fig 15. For $t = 16$ and $d = 1$ the results show a similarly good match with the model as for $t = 4$ and $d = 1$ in Fig. 13. For $d = 4$

the simulations show bigger playout buffer requirements than the model. For $d = 4$ the arrival process of the packets in the fertile trees is less regular than in the case of $d = 1$ because the nodes can be fertile in different layers. Consequently, queues build up and the waiting times of the packets ($\overline{W}_{in}$ and $\overline{W}o$) increase the convergence time.

Fig. 16 shows results for various bitrates (i.e. link utilizations), $t = 16$ and $d = 1$, and finite output link capacity $C_{out} = 512$ kbps. Increasing the bitrate decreases the convergence time even though the link utilizations increase. This conclusion follows directly from (7) and the assumption of the deterministic arrival process, as in (7) only D_b is a function of B. Nevertheless, for small bitrates shallower trees can be built if the tree maintenance algorithm increases γ^r (and hence u) for some nodes and decreases it for others. Doing so decreases the number of iterations needed, but increases $D_{tr,o}$, and hence the decrease of the convergence time itself is moderate.

Poisson arrivals: To see how the waiting times influence the performance, let us consider exponential inter-arrival times on the input links and the output links of the nodes. Although Poisson arrivals is not a worst case scenario, we can get some insight into the possible effects of the output links' utilizations on the queuing delays, and hence, the convergence time. We do not claim that the arrival process is Poisson, and our results show that it is in fact much more regular.

For Poisson arrivals we can express the mean waiting time in the input queue (assuming the queue has infinite buffer capacity) as [20]

$$\overline{W}_{in} = \frac{(B/C_{in})^2}{2\lambda_i(1 - B/C_{in})},$$

where λ_i is the arrival intensity of the packets on the input link $\lambda_i = B/b$. Similarly, the mean waiting time on the output link can be given as

$$\overline{W}_{out} = \frac{u^2}{2\lambda_o(1-u)},$$

where λ_o is the arrival intensity of the packets on the output link $\lambda_o = Bd/t/b$.

Figure 17 shows π for the scenarios considered in Fig. 14. The delays show a significant increase due to the waiting times incorporated in the model. To see which arrival model fits the actual arrival process better, we compare Figs 14, 15 and 17. The Poisson arrival process overestimates the playout buffer requirements by almost an order of magnitude. This implies that the actual arrival process of the packets is rather regular, close to deterministic. Hence the mean waiting time does not increase significantly even if nodes have nearly as many cogs as the bandwidth of their output links allows it, and the convergence time is approximately a linear function of the utilizations of the output links.

5 Conclusion

In this paper, we analyzed a peer-to-peer live streaming solution based on multiple transmission trees and FEC. We presented a mathematical model to express the packet possession probability in the overlay for the case of correlated losses. We evaluated the overlay's performance as a function of the loss probability between the peers and analyzed the asymptotic behavior of the overlay.

Based on the model and simulations, we concluded that the number of fertile trees does not influence the performance of the data transmission in the stable region of the overlay, even though the trees get deeper than optimal due to node dynamics. It influences however the stability region of the overlay and decreases the packet possession probability in the overlay in the unstable region due to the longer transmission paths. We showed that our proposal, limiting the layer spread, helps to improve the stability of the overlay, and in general, leads to improved performance.

We investigated the delay characteristics of the overlay, and concluded that the number of iterations needed for convergence is low for high packet possession probabilities, and increases as the overlay gets close to the stability threshold. We developed a model of the per-hop-delay to evaluate the effects of the playout buffer size on the data distribution performance. We showed that the delay introduced on the nodes' output links is likely to be the most important source of delay in practice, as it is proportional to the links' utilizations, and overlays tend to operate in bandwidth resource scarce environments. Our results show that FEC is the key to the stability and good performance of multi-tree-based overlay multicast. How to adjust the FEC parameters based on feedback from the peers will be subject of future research.

References

[1] V. Fodor and Gy. Dán, “Resilience in live peer-to-peer streaming,” *IEEE Communications Magazine*, vol. 45, no. 6, June 2007.

[2] X. Liao, H. Jin, Y. Liu, L.M. Ni, and D. Deng, “Anysee: Scalable live streaming service based on inter-overlay optimization,” in *Proc. of IEEE INFOCOM*, April 2006.

[3] M. Bishop, S. Rao, and K. Sripanidkulchai, “Considering priority in overlay multicast protocols under heterogeneous environments,” in *Proc. of IEEE INFOCOM*, April 2006.

[4] Y. Cui and K. Nahrstedt, “High-bandwidth routing in dynamic peer-to-peer streaming,” in *Proc. of ACM APPMS*, 2005, pp. 79–88.

[5] G. Tan and Jarvis. S.A., “A payment-based incentive and service differentiation mechanism for peer-to-peer streaming broadcast,” in *Proc. of IEEE IWQoS*, 2006, pp. 41–50.

[6] Y-W. Sung, M. Bishop, and S. Rao, “Enabling contribution awareness in an overlay broadcasting system,” in *Proc. of ACM SIGCOMM*, 2006, pp. 411–422.

[7] N. Magharei and R. Rejaie, “PRIME: Peer-to-peer Receiver drIven MEsh-based streaming,” in *Proc. of IEEE INFOCOM*, May 2007.

[8] M. Guo and M.H. Ammar, “Scalable live video streaming to cooperative clients using time shifting and video patching,” in *Proc. of IEEE INFOCOM*, 2004.

[9] S. Banerjee, S. Lee, R. Braud, B. Bhattacharjee, and A. Srinivasan, “Scalable resilient media streaming,” in *Proc. of NOSSDAV*, 2004.

[10] E. Setton, J. Noh, and B. Girod, “Rate-distortion optimized video peer-to-peer multicast streaming,” in *Proc. of ACM APPMS*, 2005, pp. 39–48.

[11] V. N. Padmanabhan, H.J. Wang, and P.A Chou, “Resilient peer-to-peer streaming,” in *Proc. of IEEE ICNP*, 2003, pp. 16–27.

[12] M. Castro, P. Druschel, A-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, “SplitStream: High-bandwidth multicast in a cooperative environment,” in *Proc. of ACM SOSP*, 2003.

[13] “OctoShape,” http://www.octoshape.com/, June 2007.

[14] Gy. Dán, V. Fodor, and I. Chatzidrossos, “On the performance of multiple-tree-based peer-to-peer live streaming,” in *Proc. of IEEE INFOCOM*, May 2007.

[15] K. Sripanidkulchai, A. Ganjam, B. Maggs, and H. Zhang, “The feasibility of supporting large-scale live streaming applications with dynamic application end-points,” in *Proc. of ACM SIGCOMM*, 2004, pp. 107–120.

[16] Gy. Dán, V. Fodor, and G. Karlsson, “On the stability of end-point-based multimedia streaming,” in *Proc. of IFIP Networking*, May 2006, pp. 678–690.

[17] Gy. Dán, I. Chatzidrossos, V. Fodor, and G. Karlsson, “On the performance of error-resilient end-point-based multicast streaming,” in *Proc. of IWQoS*, June 2006, pp. 160–168.

[18] P.B. Godfrey, S. Shenker, and Stoica. I., "Minimizing churn in distributed systems," in *Proc. of ACM SIGCOMM*, 2006, pp. 147–158.

[19] I.S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *SIAM J. Appl. Math.*, vol. 8, no. 2, pp. 300–304, 1960.

[20] J. W. Cohen, *The Single Server Queue*, North-Holland Publishing, Amsterdam, 1969.

[21] Ellen W. Zegura, Ken Calvert, and S. Bhattacharjee, "How to model an internetwork," in *Proc. of IEEE INFOCOM*, March 1996, pp. 594–602.

[22] E. Veloso, V. Almeida, W. Meira, A. Bestavros, and S. Jin, "A hierarchical characterization of a live streaming media workload," in *Proc. of ACM IMC*, 2002, pp. 117–130.

[23] K. Sripanidkulchai, B. Maggs, and H. Zhang, "An analysis of live streaming workloads on the Internet," in *Proc. of ACM IMC*, 2004, pp. 41–54.

[24] J-Y. Le Boudec and M. Vojnovic, "Perfect simulation and stationarity of a class of mobility models," in *Proc. of IEEE INFOCOM*, March 2004.

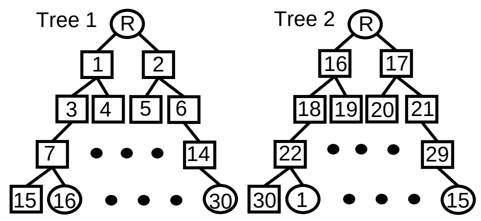

Figure 1: Overlay with $t = 2$, $d = 1$, $N = 30$, $m = 2$ with unlimited layer spread. Square indicates that the node is fertile.

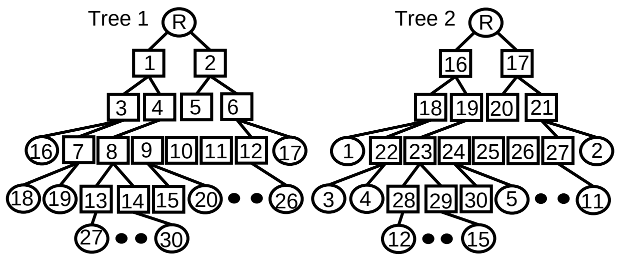

Figure 2: Overlay with $t = 2, d = 1, N = 30, m = 2$ and with layer spread limit $\delta = 2$. Square indicates that the node is fertile.

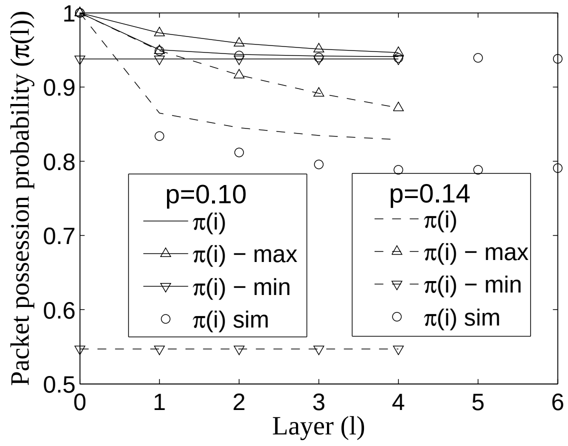

Figure 3: $\pi(l)$ vs. l for $m = 50$, $t = n = 4$, $c = 1$, $\overline{N} = 10^4$.

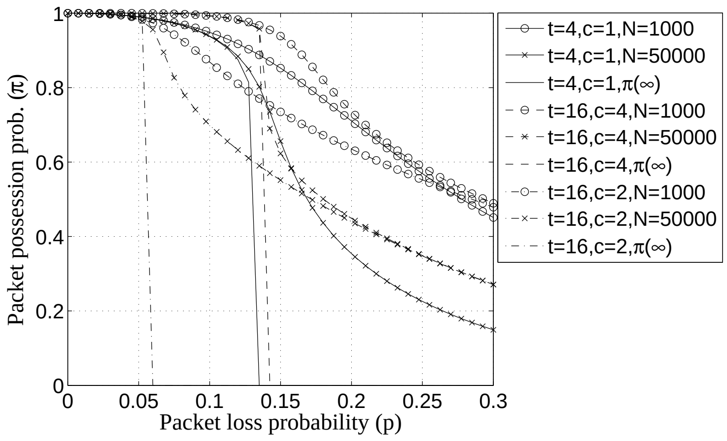

Figure 4: π vs. p for $m = 50$, $n = t$, $d = 1$.

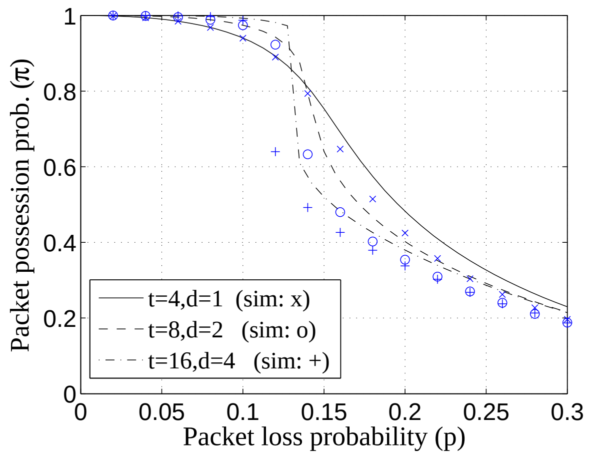

Figure 5: π vs. p for $m = 4$, $n = t$, $k/n = 0.75$.

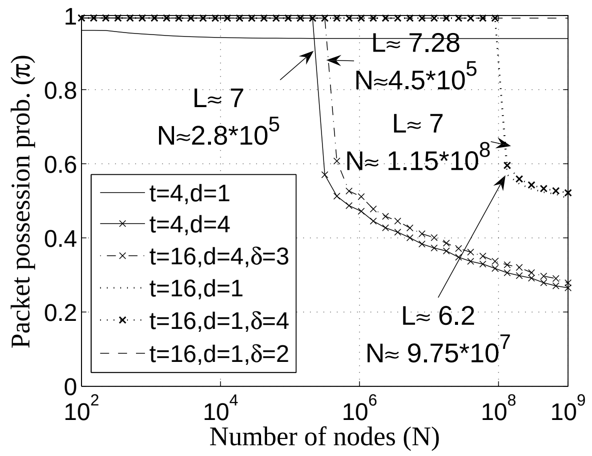

Figure 6: π vs. $\overline{N}$ for $n = t$, $k/n = 0.75$, $p = 0.10$.

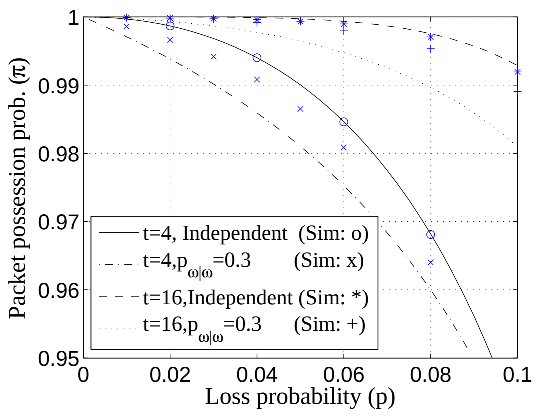

Figure 7: π vs. p for independent and correlated losses, $m = 50$, $n = t$, $t/d = 4$, $k/n = 0.75$, $\overline{N} = 10^4$.

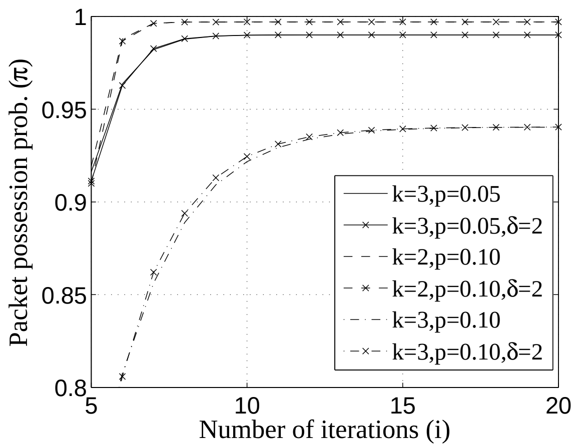

Figure 8: π vs. i for $t = 4$, $d = 1$, $n = 4$, $m = 50$, $\overline{N} = 10^4$.

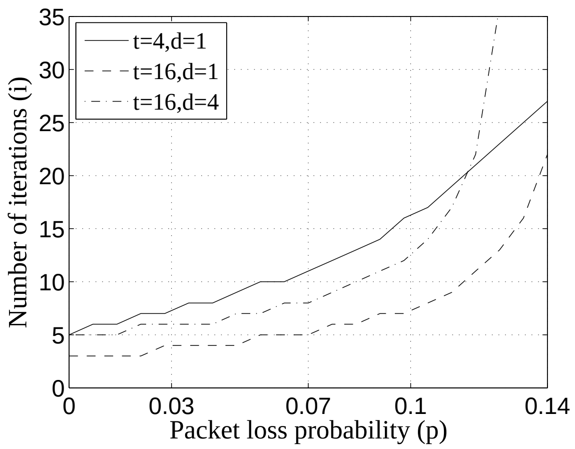

Figure 9: i vs. p for $t = n$, $m = 50$, $\overline{N} = 10^4$.

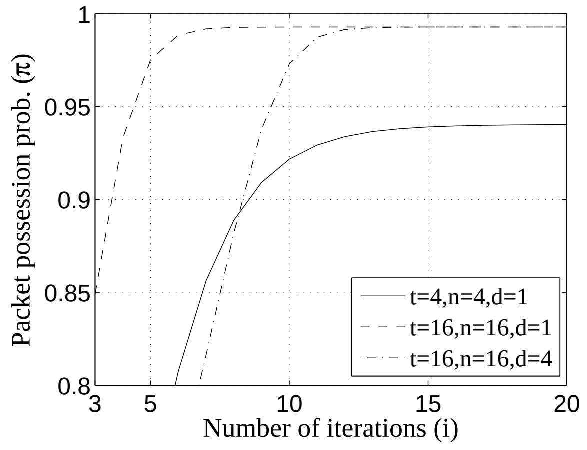

Figure 10: π vs. i for $p = 0.1$, $m = 50$, $k = 0.75n$, $\overline{N} = 10^4$.

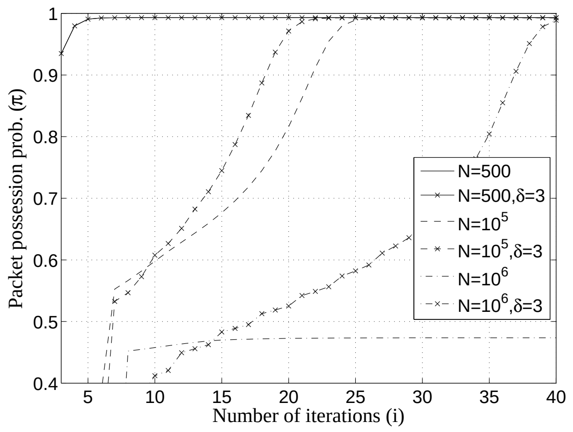

Figure 11: π vs. i for $p = 0.1$, $t = n = 16$, $d = 4$, $k = 12$, $m = 50$.

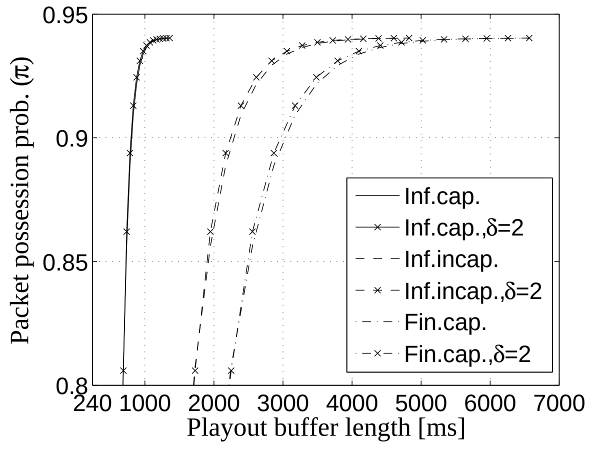

Figure 12: π vs. delay for $p = 0.1$, $t = 4$, $d = 1$, $n = 4$, $m = 50$, $\overline{N} = 10^4$. Deterministic arrivals.

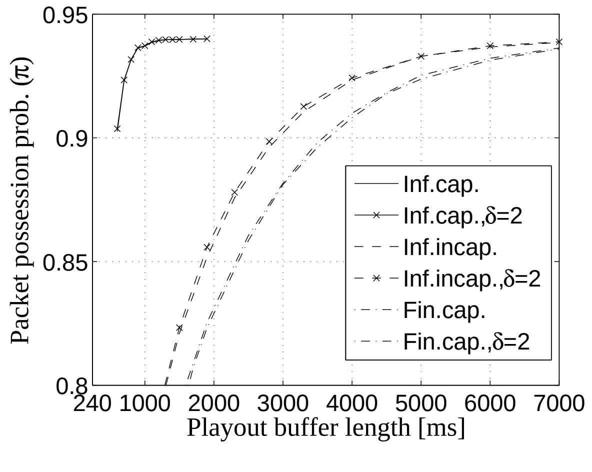

Figure 13: π vs. delay for $p = 0.1$, $m = 50$, $k = 0.75n$, $\overline{N} = 10^4$. Simulations.

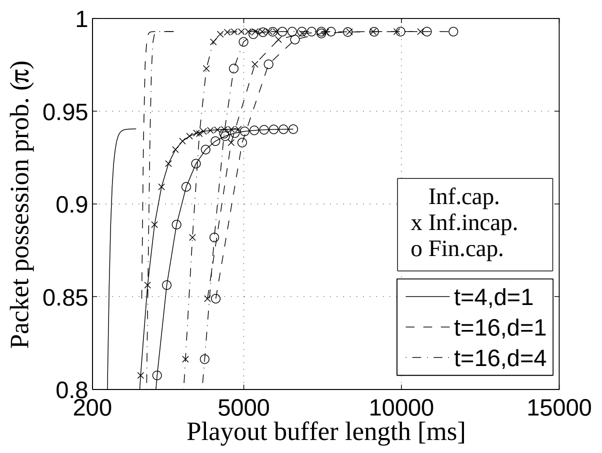

Figure 14: π vs. delay for $p = 0.1$, $m = 50$, $k = 0.75n$, $\overline{N} = 10^4$. Deterministic arrivals.

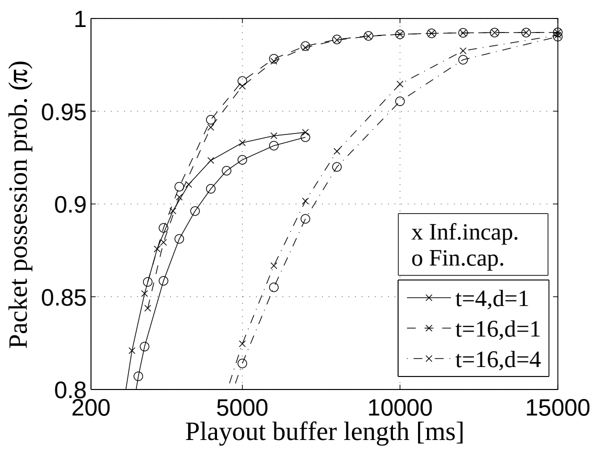

Figure 15: π vs. delay for $p = 0.1$, $m = 50$, $k = 0.75n$, $\overline{N} = 10^4$. Simulations.

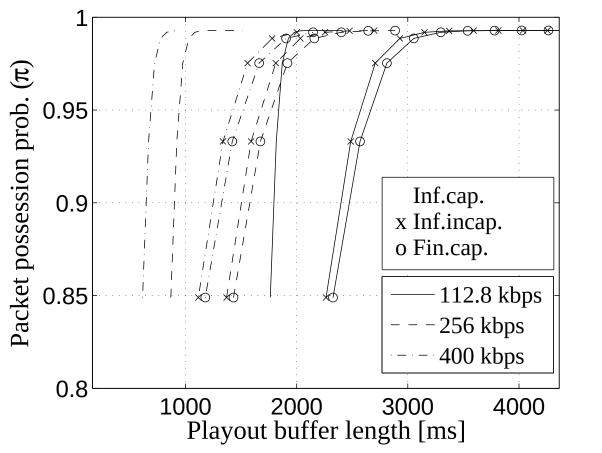

Figure 16: π vs. delay for $p = 0.1$, $m = 50$, $t = 16$, $d = 1$, $k = 0.75n$, $\overline{N} = 10^4$. Deterministic arrivals.

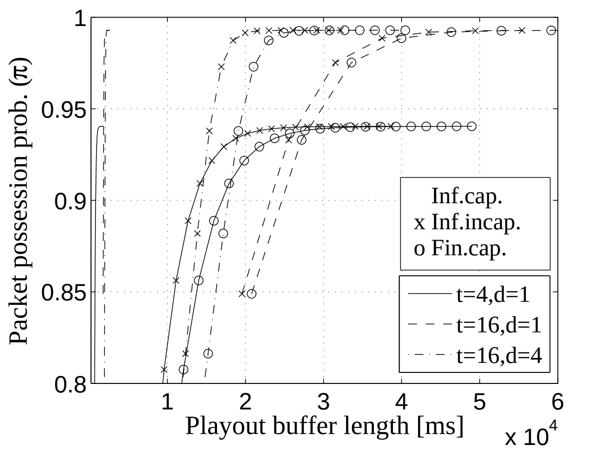

Figure 17: π vs. delay for $p = 0.1$, $m = 50$, $k = 0.75n$, $\overline{N} = 10^4$. Poisson arrivals.