Signal Processing: *Image Communication* 23 (2008) 754–768

Contents lists available at ScienceDirect

Signal Processing: *Image Communication*

journal homepage: www.elsevier.com/locate/image

Distributed media rate allocation in multipath networks ☆

Dan Jurca *, Pascal Frossard

Ecole Polytechnique Fédérale de Lausanne (EPFL), Signal Processing Institute, CH-1015 Lausanne, Switzerland

ARTICLE INFO

Article history:
Received 28 March 2008
Received in revised form
9 September 2008
Accepted 10 September 2008

Keywords:
Video streaming
Distributed rate allocation
Optimal path selection
Distortion optimized video streaming
Multipath media transmission

ABSTRACT

The paper addresses the distributed path computation and rate allocation problems for video delivery over multipath networks. The streaming rate on each path is determined such that the end-to-end media distortion is minimized, when a media client aggregates packets received via multiple network channels to the streaming server. In common practical scenarios, it is, however, difficult for the server to have the full knowledge about the network status. Therefore, we propose here a distributed path selection and rate allocation algorithm, where the network nodes participate to the optimized path selection and rate allocation based on their local view of the network. This eliminates the need for end-to-end network monitoring, and permits the deployment of large scale rate allocation solutions. We design a distributed algorithm for optimized rate allocation, where the media client iteratively determines the best set of streaming paths, based on information gathered by network nodes. Each intermediate node then forwards incoming media flows on the outgoing paths, in a distributed manner. The proposed algorithm is shown to quickly converge to the rate allocation that provides a maximal quality to the video client. We also propose a distributed greedy algorithm that achieves close-to-optimal end-to-end distortion performance in a single pass. Both algorithms are shown to outperform simple heuristic-based rate allocation approaches for numerous random network topologies. They offer an interesting solution for media-specific rate allocation over large scale multipath networks.

1. Introduction

As the internet is far from providing any widely deployed guarantee of service, efficient media streaming strategies have to be devised to cope with the weaknesses of the network infrastructure, and provide an acceptable quality to multimedia applications. Multipath streaming emerged lately as an effective solution to overcome some limitations of lossy internet paths [6,15]. It offers an increase in streaming bandwidth by balancing the load over multiple network paths between the media server and the client. It also provides means to limit packet loss effects, when combined with error resilient streaming strategies and scalable encoding capabilities. Multipath streaming can be deployed in content delivery networks, overlay networks or wireless and peer-to-peer scenarios, where a client has access to the media sources simultaneously through multiple network paths.

This paper addresses the problem of media-specific rate allocation for streaming applications in multipath networks.

For given network parameters, an optimal set of transmission paths are selected, along with their respective transmission rate, such that the decoded media quality is maximized. However, path selection performed exclusively at the server or at the client generally requires end-to-end network monitoring, as well as the knowledge of the complete network status at one point of the

☆ This work has been supported by the Swiss National Science Foundation, under Grant PP-002-68737.

* Corresponding author.
E-mail addresses: dan.jurca@gmail.com, jurca@docomolab-euro.com (D. Jurca), pascal.frossard@epfl.ch (P. Frossard).

0923-5965/$ - see front matter © 2008 Published by Elsevier B.V.
doi:10.1016/j.image.2008.09.002

topology. This unfortunately limits the implementation of such algorithms to small-scale network scenarios. Therefore, we propose in this paper, a distributed solution where intermediate network nodes that are capable of handling application-level information, participate to the path selection and rate allocation algorithm based on their local view of the network. Network nodes become actively involved in the media delivery between a streaming server and a media client; it permits to deploy efficient video distribution applications in large scale distributed architectures such as mesh or peer-to-peer networks.

We build on our prior work [10] that describes a server-driven framework for the analysis of joint path and rate allocation in multipath video streaming. We, however, bypass the limitations of centralized approaches, and we propose here novel algorithms for computing the streaming paths and the streaming rates in a distributed manner. We consider a network model composed of multiple flows between the client and the streaming server, e.g., overlay networks or networks with path diversity. The intermediate network nodes together report the resources available for the streaming session. Based on this information, the client determines the best path selection and rate allocation, and generates flow reservation requests to the intermediate network nodes and the streaming server. The client-based flow reservation is then accommodated within the network on a node-by-node basis. The joint path selection and rate allocation performs iteratively, until all intermediate nodes converge to a (unique) optimal solution. The server finally adapts the media source rate accordingly by scalable coding, or packet filtering for example.

The new distributed path selection strategy permits to relax the assumption of full network status knowledge at the server and to eliminate the need for complex end-to-end network monitoring strategies. We design a path selection algorithm that quickly converges to the optimal video rate allocation solution. In parallel, we propose a fast, one-step algorithm, which provides a close-to-optimal solution. The performance of both algorithms are analyzed in details and compared to simple heuristic-based approaches. Thanks to the optimal media-specific allocation, the proposed algorithms clearly outperform other strategies based on pure network metrics. Thanks to their distributed nature and to the media-specific considerations, these algorithms offer effective solutions to the media streaming rate allocation problem in medium to large scale multipath networks.

The rest of this paper is organized as follows. Section 2 discusses the related work and motivates the need for distributed rate allocation solutions. Section 3 describes in detail the streaming scenario considered in this paper, and presents the rate allocation optimization problem. We present our distributed solutions in Section 4 and we analyze the characteristics of the proposed algorithms in Section 5. Extensive simulation results are finally presented in Section 6 for numerous network topologies.

2. Related work

This paper addresses the multipath routing problem from a media application perspective. The process of selecting the paths for transmission and their respective rate allocation, targets an improved streaming experience measured in terms of video distortion. Several prior works have discussed the path selection problem based on pure network metrics. These works generally lead to suboptimal solutions in terms of media quality. For example, numerous routing algorithms have been proposed in order to optimize a given network QoS metric [28], to improve the performance of TCP over wireless ad-hoc networks [20], to discover multiple available network paths to one source [32], or to optimize the network resource allocation in overlay multicasts [3,25]. In addition, the authors of [33] adapt the DSR protocol for ad-hoc networks to provide multiple viable paths for multimedia transmissions. However, none of these works specifically considers the multimedia application characteristics in the routing decisions. They rather rely on routing algorithms that find the best path (or set of paths), given some established network metrics. While this may be optimal in terms of network utilization, it is, however, suboptimal from the point of view of the quality of service for the media streaming application. In 30–80% of the cases, the best paths found by classic routing algorithms are suboptimal from a media perspective [26].

At the same time, several works have investigated the problem of multipath streaming, as a way to increase the multimedia quality of service on lossy network infrastructures. Nevertheless, most of the research work dedicated to multipath streaming focuses on the streaming process itself (media scheduling aspects), but generally not towards finding which paths should ideally be used for the streaming application, for a given network topology. More specifically, the multipath problem is addressed in the case of media streaming in [21] for a multicast scenario. The authors present a FEC scheme combined with server diversity and a packet scheduling mechanism, which intends to minimize the cumulative distortion of individual erroneous video packets. Multistream coding, combined with multipath transmission, has been presented in [17] as a solution to fight against network errors in an ad-hoc network environment. The authors of [1] multiple path streaming scenario for the transmission of video sequences encoded in multiple descriptions. Other works in distributed video streaming [13,35,22] deal with resource allocation and scheduling on multiple, a priori chosen streaming paths, with the final goal of minimizing the overall distortion perceived by the media clients. All these works rely on a given set of transmission paths, and try to optimally exploit these network resources. However, none of these specifically targets the optimal choice of the streaming paths or the rate allocation problem for improving the multimedia quality of service. The work presented in [30] addresses the problem of choosing the best paths from a media perspective. However, it only investigates the efficiency of path switching schemes from the media application point

of view, without analyzing the benefits of multipath streaming.

In this work, we address the problem of joint selection of network paths and rate allocation, such that the end-to-end media distortion is minimized. In addition to considering multipath strategies that have been shown to improve streaming performances, our work also innovates by proposing a distributed solution for path computation. It alleviates the need for expensive end-to-end path monitoring systems, and thus can be deployed in large scale network scenarios. Our streaming framework is quite generic and applies to any streaming system that obeys an additive rule for the aggregated transmitted rate and packet loss probabilities. The optimal routing and rate allocation decision determines the best usage of end-to-end transmission paths, so that the media distortion is minimized when network flows are aggregated at the decoder.

3. The distributed multipath rate allocation problem

3.1. Network and video model

We consider that the media streaming application is deployed on a large scale network, e.g., overlay networks or networks with path diversity. The network is modelled as a fully connected, directed acyclic graph $G(V,E)$, between the streaming server S and the client C (Fig. 1). V is the set of nodes in the network, and E is the set of links. Each node $N_i \in V$ has a *local view* $\mathcal{N}_i = \{I_i, O_i\}$ of the network topology, where $I_i \subseteq E$ and $O_i \subseteq E$ represent the sets of incoming and respectively outgoing network links for the node N_i. Each link $L_u \in E$ has two associated positive metrics:

- the available bandwidth $\rho_u > 0$ and
- the average packet loss probability $p_u \in [0,1]$, assumed to be independent of the streaming rate.

We define P_C^i, $1 \leqslant i \leqslant n$, as an end-to-end path between S and C in G, with parameters b_C^i and p_C^i being the end-to-end bandwidth and the loss probability, respectively, and

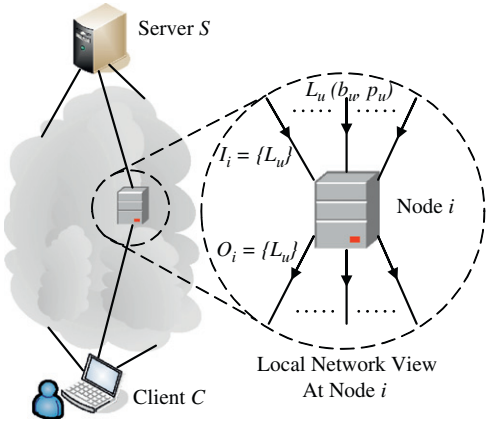

Fig. 1. Multipath network scenario and network view at Node N_i.

n the total number of distinct paths. A flow[1] transmitted on path P_C^i has a streaming rate $r_C^i \leqslant b_C^i = \min_{L_u \in P_C^i}(\rho_u)$, and is affected by the loss probability $p_C^i = 1 - \Pi_{L_u \in P_C^i}^{L_u \in P_C^i}(1 - p_u)$.

We consider applications with stringent delay constraints and no possibility for packet retransmission. Typically, we use UDP streaming strategies that are generally preferred to TCP solutions in many streaming scenarios. The video quality depends in this case on the actual streaming/encoding rate and transmission loss probabilities. We consider that the end-to-end media distortion can be computed as the sum of the source distortion and the channel distortion. It is commonly admitted that the quality experienced at the client, depends on both the distortion due to a lossy encoding of the media information and the distortion due to losses experienced in the network (see for example [5,29]). For a given video encoder, the source distortion D_S is mostly driven by the encoding rate R (also called streaming rate in this paper), and the media sequence content, whose characteristics influence the rate-distortion characteristics of the encoder. The channel distortion D_L is dependent on the average loss probability e sustained by video information, and the sequence characteristics. It is roughly proportional to the number of video entities (e.g., frames) that cannot be decoded, and the loss probability e corresponds to the actual video packet loss ratio when video frames are encapsulated into distinct network packets. The average end-to-end distortion can thus be written as

$$D = D_S + D_L = \alpha \mathscr{R}^{\xi} + \beta \in \tag{1}$$

where α, $\beta \in \Re^+$ and $\xi \in [-1,0]$ are parameters that depend on the video sequence. In the above multipath streaming scenario, the streaming rate can simply be written as the sum of the rates of the different flows

$$\mathscr{R} = \sum_{i=1}^{n} r_C^i.$$

At the same time, when the loss processes on different paths are independent, the overall loss probability becomes

$$\in = \frac{\sum_{i=1}^{n} p_C^i r_C^i}{\sum_{i=1}^{n} r_C^i}.$$

The average end-to-end distortion model is a simple and general approximation, suitable for most common streaming strategies where the number of packets per frame is independent of the encoding rate, which can be adjusted to the network offered streaming rate, e.g., through scalable encoding. Note that the actual video loss process is likely to present a low correlation, due to the usage of multiple paths. Under the given network assumptions, the video distortion metric becomes quite insensitive to the actual link error model, and is mostly influenced by the average loss probabilities. A validation of this model through video experiments can be found in Ref. [10].

[1] Throughout this paper, the terms flow and end-to-end network path are used interchangeably.

In the remainder of this section, we first review briefly the results for the multipath media rate allocation problem in centralized systems. The relaxation of the assumption about full knowledge of the network then leads to a novel problem formulation for distributed path selection in multipath video streaming. The solution of this problem leads to the rate allocation algorithms described in this paper.

3.2. Optimal rate allocation

This section briefly overviews the solution to the optimal rate allocation, when the server has full knowledge about the status of the network. In our previous work [10], we have derived the analytical rules that allow deriving the optimal solution to the joint path selection and rate allocation problem for a given video stream, with a simple algorithm whose complexity is linear in the number of available end-to-end network paths. Namely, once the parameters of all paths $P_C{}^i$ are known by S, the three following theorems lead to the optimal greedy rate allocation solution in any loop-free flow-equivalent network graph. A *flow-equivalent graph* G denotes a network topology where the maximum bandwidth offered to the application (e.g., the maxflow of G) does not depend on the choice of transmission paths. Flow-equivalent networks represent most of the typical topologies encountered in practice, where bottleneck links typically lie on the edge of the network, or between domains. Interested readers are referred to Ref. [9] for ample discussions and proofs of the following theorems. They are given here for the sake of completeness, as they are used in the distributed algorithm proposed in the next section.

Theorem 1 (*on–off flows*). *Given a flow-equivalent network graph G with independent flows $\mathscr{F}_C^i$ having rates $r_C^i \in [0, b_C^i]$ and a distortion metric as defined in Eq.* (1)*, the optimal solution of the rate allocation problem when all the paths are disjoint, lies at the margins of the value intervals for all r_C^i, i.e., the optimal value of r_C^i is either 0 or b_C^i, $\forall i: 1 \leqslant i \leqslant n$.*

Theorem 2 (*parameter decoupling*). *Given a flow-equivalent network graph G with independent flows $\mathscr{F}_C^i$ having rates $r_C^i \in [0, b_C^i]$ and a distortion metric as defined in Eq.* (1)*, the structure of the optimal rate allocation is $\Phi^* = [b_C^1, b_C^2, \ldots, b_C^i, 0, 0, \ldots 0]$.*

Theorem 3 (*bottleneck bandwidth sharing*). *Let L_u be a bottleneck link for the set of paths $\mathscr{B}_u = \{P_C^k\}$ in G. The bottleneck link bandwidth ρ_u shall be shared among paths P_C^k in a greedy way, starting with the path affected by the lowest loss probability.*

When the characteristics of all paths $P_C{}^i$ are known by the server S, Theorems 1–3 offer the guidelines for a complete optimal path selection and rate allocation solution. They show that the optimal rate allocation can be achieved by a greedy path selection algorithm that starts with the paths affected by the smallest end-to-end loss probability. At the same time, the rate of bottleneck links that are shared by multiple network paths should also be split in a greedy manner among media flows. Once a path P_C^i is chosen for transmission, it is optimal to stream at rate $r_C^i = b_C^i$, from the media application perspective.[2] Based on these rules, the optimal path selection and rate allocation can be achieved by a greedy algorithm, which provides a low complexity solution to media-specific resources optimization in flow-equivalent networks. We present now novel distributed mechanisms for computing the available end-to-end paths on the network graph, so that the strong assumption of full network knowledge at the streaming server can be relaxed. We eventually build on Theorems 1–3 in order to compute the optimal rate allocation on these paths.

3.3. Distributed optimization problem

We now formalize the distributed path selection and rate allocation problem addressed in this paper. When no single node $N_i \in V$ (including S), is aware of the entire network topology G, we want to find the optimal path selection and flow rate allocation that minimizes the overall distortion D at the client. Under the assumptions that the streaming rate can be controlled and that the packet loss rate is independent of the streaming rate, the server S eventually adapts the video encoding rate to the aggregate rate of the available network paths used for streaming, and to the loss processes experienced on these paths.[3] The optimization problem can be formulated as follows:

3.3.1. Distributed multimedia rate allocation problem (DMMR)

Given the flow-equivalent network graph $G(V, E)$ whose links L_u have a maximal bandwidth ρ_u and an average loss ratio p_u, given the node local views $\mathscr{N}_i, \forall N_i \in V$ and given the video sequence characteristics $\Gamma = (\alpha, \beta, \xi)$, find the complete set of end-to-end paths P_C^i, $1 \leqslant i \leqslant n$ and the optimal rate allocation $\vec{\mathscr{R}}^* = [r_C^1, \ldots, r_C^n]^*$ that minimizes the distortion metric D

$$\vec{\mathscr{R}}^* = \underset{\vec{\mathscr{R}}}{\arg\min}\, D = \underset{\vec{\mathscr{R}}}{\arg\min}(\alpha \mathscr{R}^{\xi} + \beta \in) \tag{2}$$

under the constraints

$$r_C^i \leqslant b_C^i, \quad \forall P_C^i, \quad 1 \leqslant i \leqslant n$$
$$\sum_{P_C^i : L_u \in P_C^i} r_C^i \leqslant \rho_u, \quad \forall u \text{ such that } L_u \in E$$

where $\vec{\mathscr{R}}$ represents the set of possible rate allocations on $G(V, E)$, $\mathscr{R} = \sum_{i=1}^{n} r_C^i$ and $\varepsilon = \frac{\sum_{i=1}^{n} p_C^i \cdot r_C^i}{\sum_{i=1}^{n} r_C^i}$.

[2] Note that in our developments, we assume that b_C^i is the total fair share of bandwidth allocated by the network for the streaming application on path P_C^i, and that the streaming flows do not suffer from self-congestion.

[3] For a discussion of a more general transmission error process, we refer the interested reader to [12]; [27].

4. Distributed rate allocation

4.1. Distributed path computation

We present in this section two algorithms for distributed path selection and rate allocation. The algorithms differ in the computation of the paths between the server S and the client C. Before describing in detail the distributed path computation and rate allocation strategies, we briefly introduce the notation and assumptions necessary to their presentation. Recall that every node $N_i \in V$ has only a local view of the network topology, denoted by $\mathcal{N}_i = \{I_i, O_i\}$. I_i and O_i are the sets of incoming and respectively outgoing links to/from N_i. We assume that N_i possesses an estimate of the bandwidth ρ_u and loss probability p_u on all the outgoing links (i.e., $\forall L_u \in O_i$).

Let P_i^k denote a path connecting the node N_i to the server. In addition to maximal bandwidth b_i^k and loss probability p_i^k, a path is characterized by two decisions flags that are used by the distributed rate allocation algorithms. The flag f^k is a path reservation flag that can only be set or reset by the client C, respectively the server S, and the flag d^k is a decision flag that can be updated by any intermediate node on the path P_i^k. While f^k is used to advertise the network flows requested by the client C, d^k is used to signal the feasibility of a requested flow at an intermediate node.

We denote by $\Pi_i = \{P_i^k\}$ the set of all distinct paths between the server S and the node N_i. Note that two distinct paths P_i^k and P_i^l may not necessarily be fully disjoint, as they may share one or more network links. Without loss of generality, we assume that the paths in Π_i are ordered according to the increasing value of the path loss probabilities p_i^k. Let finally $\Pi_i^u \subseteq \Pi_i$ be the set of distinct paths between the server S and the node N_i, which share the incoming link $L_u \in I_i$.

End-to-end paths between the server and the client are then built in a distributed manner, since no node has the full knowledge of the network status. These paths are computed by path extension, which is performed independently at each network node. We choose the notation '$\rightarrow$' to represent the path extension operator that adds a link $L_u \in O_i$ leaving node N_i, to an incoming path $P_i^k \in \Pi_i$. In other words, if link L_u connects nodes N_i and N_j, we can write $P_j^l = P_i^k \rightarrow L_u$, with $P_j^l \in \Pi_j^u$ and $P_i^k \in \Pi_i$. We can compute the bandwidth and loss probability parameters for the extended path $P_j^l = P_i^k \rightarrow L_u$, respectively, as $b_j^l = \min(b_i^k, \rho_u)$ and $p_j^l = 1-(1-p_i^k)(1-p_u)$.

We propose two different methods for distributed path computation (employed by the two proposed algorithms), which respectively constructs all the possible paths, or builds them in a greedy manner. Formally, the two path extension rules can be stated as follows.

Rule 1: Each incoming path $P_i^k \in \Pi_i$ at node N_i is extended towards all the outgoing links $L_u \in O_i$.

If the set of outgoing links directly connect N_i to several nodes N_j, the set of extended paths at node N_i can be written as $\Omega_i = \{P_j^l = P_i^k \rightarrow L_u | P_i^k \in \Pi_i, L_u \in O_i\}$. The subset of the extended paths that borrow the particular outgoing link L_u is written as $\Omega_i^u = \{P_j^l = P_i^k \rightarrow L_u | P_i^k \in \Pi_i\}$. All paths with null bandwidth are obviously omitted. It is easy to see in this case that $|\Omega_i^u| = |\Pi_i|$ and that $|\Omega_i| = |\Pi_i||O_i|$. The size of the set is multiplicative in the number of incoming flows and in the number of outgoing links [16]. It has to be noted that resource allocation for flows in Ω is constrained by the available bandwidth on joint bottleneck links, and that all the paths may not be used simultaneously at their full transmission bandwidth.

Rule 2: The incoming paths $P_i^k \in \Pi_i$ at node N_i, taken in order of increasing loss probability p_i^k are extended towards the outgoing links $L_u \in O_i$, taken in decreasing order of reliability. Similarly to a water-filling algorithm, the total outgoing bandwidth is greedily allocated to the set of incoming paths, until all the incoming paths are extended, or until no more bandwidth is available.

When the sets of outgoing links and the incoming paths are both ordered along the increasing values of loss probability, the set of extended paths at node N_i can be written as

$$\Gamma_i = \left\{ P_j^l = P_i^k \rightarrow L_u \left| \sum_{\mu=1}^{u} \rho_\mu > \sum_{v=1}^{k-1} b_i^v \text{ and } \sum_{\mu=1}^{u-1} \rho_\mu < \sum_{v=1}^{k} b_i^v \right. \right\}$$

The subset of the paths in Γ_i that borrow the outgoing link L_u is denoted Γ_i^u. Note that in this case, simultaneous resource allocation for all flows in Γ_i, is feasible on G.

Based on the distributed path computation that follows either Rule 1 or 2, we now describe two rate allocation strategies called, respectively, Algorithms 1 and 2. Then we present both an optimal and a greedy rate allocation algorithm for distributed multipath media streaming.

4.2. Distributed path selection and rate allocation

The distributed path computation and rate allocation algorithms proceed first by determining the paths available between the server and the client. Then they reserve paths according to the optimal allocation computed by the client. They proceed in two phases, the path discovery and the path reservation phases, respectively. To this aim, control messages are exchanged at the application layer between the server S and the client C, and forwarded by the intermediate nodes, as illustrated in Fig. 2. We assume the existence of a bidirectional control channel between any two nodes in G that are connected by a network segment L_u.

The server initiates the first phase of the algorithms by sending path discovery messages $Path^u$ on all outgoing links. The messages are forwarded by the intermediate nodes in the considered network overlay, on the control channel associated with link L_u. At each intermediate node, the *Path* messages contain the information (b_i^k and p_i^k) related to every possible flow between the server and node N_i, along with information related to previously successfully reserved flows. The node then extends the path according to Rule 1 or 2 (in the case of Algorithm 1 or 2, respectively), and forwards path discovery message $Path^u$ that contains information about the paths that borrow the link L_u. Depending on the path extension strategy that uses either Rule 1 or 2, the client eventually receive information about all possible paths, or

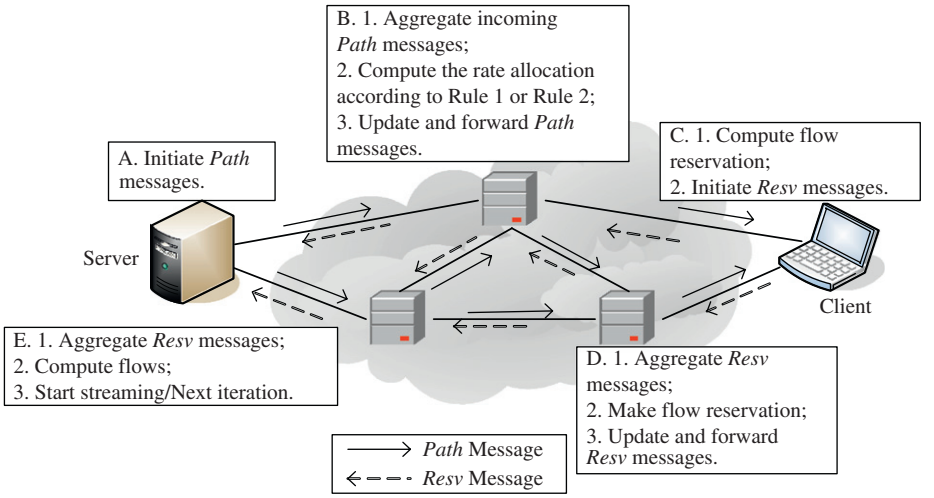

Fig. 2. Distributed path selection and reservation.

respectively about only a subset of them. In the latter case, the paths are computed in a greedy manner along the decreasing reliability of the end-to-end channels.

Upon reception of path discovery messages, the client C computes the optimal path selection Π^*_C using the Theorems 1–3, and the information it gets from the nodes about end-to-end paths. It should be noted that these theorems greatly simplify the rate allocation in flow-equivalent networks. They indeed state that paths should either be used at their full bandwidth, or simply dropped. The client then initiates the second phase of our algorithms, by sending path reservation messages, $Resv^u$, which are forwarded by the network nodes N_i to the server, on the backward control channel associated with link L_u.[4] A path reservation message $Resv^u$ contains information about the flow(s) that should be reserved on link L_u for the streaming session (e.g., requested rate b^k_C, end-to-end loss probability p^k_C and flags f^k and d^k, which are both set to 1 by C). However, there is no guarantee that all paths in Π^*_C, can be accommodated simultaneously. Once all *Resv* messages are received at node N_i (one for each outgoing link), the node N_i attempts to greedily allocate the bandwidth for the requested flows ($d^k = f^k = 1$) on the outgoing links, following the order of increasing loss probability p^k_C. It eventually marks the flows that cannot be reserved at the requested rate b^i_C, by setting the flag $d^k = 0$. Once a valid subset of paths $\Pi^* \subseteq \Pi^*_C$ is successfully reserved by S (i.e., all d^k flags are set to 1), the nodes update their local view of the network, $\mathcal{N}'_i = \mathcal{N}_i \backslash \Pi^*$, and new path discovery messages are issued. The client aggregates information about the residual network resources, and updates the path selection Π^*_C accordingly. The process is iterated until convergence to the final rate allocation, which is reached when all flows reserved by C can be accommodated by the network at the requested rate b^k_C.

The distributed path selection and rate allocation algorithms illustrated in Fig. 2 are finally summarized in Algorithms 1 and 2, where the left-hand side and right-hand side columns, respectively, correspond to the path discovery and path extensions phases. Initially, both algorithms start at the server side, with step 4. The algorithms differ in the path extension rule (step 3 in the bottom left block).

The path extension rule directly controls the convergence to the stable rate allocation, but also the quality of the rate allocation solution. Comprehensive information about end-to-end paths as created by Rule 1 permits to reach an optimal rate allocation, but possibly at the expense of several iterations of the path reservation schemes. The algorithm, however, converges in a small number of rounds to a feasible solution,

Algorithm 1. Distributed path selection and rate allocation algorithm-optimal

Server S:
Upon receive $Resv^u$, $\forall L_u \in O_S$:
1. compute Π^*_C based on flags f^k;
2. update Π^* based on flags d^k;
3. if $\Pi^* = \emptyset$ or $\Pi^* = \Pi^*_C$, return Π^*;.
4. else update network view $\mathcal{N}'_S$ send $Path^u$, $\forall L_u \in O_S$.

Node N_i:
Upon receive $Resv^u$, $\forall L_u \in O_i$:
1. $\forall$ paths $P^k_i \in \{P^k_i\} | P^k_i \rightarrow L_u \in Resv^u \backslash \Pi^*$: set $d^k = 0$ if $b^k_C > \rho'_u$, where the available output bandwidth ρ'_u is updated according to a greedy allocation;
2. send $Resv^v$, $\forall L_v \in I_i$.

Node N_i:
Upon receive $Path^u$, $\forall L_u \in I_i$:
1. update network graph $\mathcal{N}'_i$;
2. compute available paths Π_i according to $\mathcal{N}'_i$;

Client C:
Upon receive $Path^u$, $\forall L_u \in I_C$:
1. compute the set of available paths Π_C;
2. compute the optimal allocation Π^*_C from Π_C;

[4] Due to practical implementation considerations, an empty *Resv* message should be sent even on links that do not contain any reserved flow. Alternatively, timeouts should be implemented at each intermediate node.

3. compute extended paths Ω_i resp. Γ_i, $\forall L_v \in O_i$, according to Rule 1;
4. send discovery messages $Path^u$, $\forall L_v \in O_i$.

3. $\forall P_C^k \in \Pi^*_C$, set $f^k = d^k = 1$;
4. send reservation messages $Resv^v$, $\forall L_v \in I_C$.

Algorithm 2. Distributed path selection and rate allocation algorithm-greedy

Server S:
Upon receive $Resv^u$, $\forall L_u \in O_S$:
1. compute Π^*_C based on flags f^k;
2. update Π^* based on flags d^k;
3. if $\Pi^* = \emptyset$ or $\Pi^* = \Pi^*_C$, return Π^*;
4. else update network view $\mathcal{N}'_S$ send $Path^u$, $\forall L_u \in O_S$.

Node N_i:
Upon receive $Resv^u$, $\forall L_u \in O_i$:
1. $\forall$ paths $P_i^k \in \{P_i^k\} | P_i^k \rightarrow L_u \in Resv^u \backslash \Pi^*$: set $d^k = 0$ if $b_C^k > \rho'_u$, where the available output bandwidth ρ'_u is updated according to a greedy allocation;
2. send $Resv^v$, $\forall L_v \in I_i$.

Node N_i:
Upon receive $Path^u$, $\forall L_u \in I_i$:
1. update network graph $\mathcal{N}'_i$;
2. compute available paths Π_i according to $\mathcal{N}'_i$;
3. compute extended paths Ω_i, resp. Γ_i $\forall L_v \in O_i$, according to Rule 2;
4. send discovery messages $Path^v$, $\forall L_v \in O_i$.

Client C:
Upon receive $Path^u$, $\forall L_u \in I_C$:
1. compute the set of available paths Π_C;
2. compute the optimal allocation Π^*_C from Π_C;
3. $\forall P_C^k \in \Pi^*_C$, set $f^k = d^k = 1$;
4. send reservation messages $Resv^v$, $\forall L_v \in I_C$.

given the network graph G. The Rule 2 constructs only a limited subset of end-to-end network paths, given a greedy forwarding solution at each intermediate node N_i. It leads to a quicker computation of the solution, which may however be suboptimal. Both algorithms are analyzed in Section 5 and their performance is compared in Section 6.

5. Analysis and discussion

5.1. Convergence properties

This section proposes an analysis of the path selection and rate allocation algorithms introduced in the previous section. Under the assumption that the network is stable during the execution of our algorithms, we derive hard bounds on the convergence of the rate allocation towards the optimized solution. Observe that one iteration of the algorithms requires one complete message exchange between S and C, on the available paths. Hence, the time required by one round is in the order of the round-trip time (RTT) of the slowest paths in the network. The computations at intermediate nodes and at S and C are trivial and their duration can be neglected. The rate allocation algorithms generally converge in a very small number of steps (as shown in the next section), so that the assumption about the stability of the network in terms of average bandwidth and loss probability of the network links is usually valid. Since the total number of paths is quite small in general [10], the algorithms reach a stable solution in a convergence time corresponding to only a small number of RTTs. This is equal to the number of rounds of the iterative path selection that are needed for convergence, and the average link characteristics are likely to stay unchanged during that period.

In order to derive exact bounds on the performance of our algorithms, we further assume that the control channel is reliable, and that nodes are synchronized, i.e., any node receives all dedicated control packets in a bounded time interval. Note that perfect synchronization is not crucial for the design of the proposed algorithms, which can work with looser synchronization. Loose node synchronization can be achieved by employing separate synchronization protocols [18]. Most works addressing decentralized systems [24] assume loose node synchronization in order to derive bounds on protocol performance.

We consider first the Algorithm 1, which uses Rule 1 for path extension: the client has a complete view of end-to-end paths to compute the path selection. We show that the Algorithm 1 converges to the optimal solution in one round if paths are disjoint. Then, we show that in the worst case, one round of the algorithm reserves at least the path with the lowest loss probability. Consequently, the Algorithm 1 terminates in a finite number of rounds. We now formally prove these three properties.

Property 1. *If the paths requested by C do not share any bottleneck joint link L_u,* Algorithm 1 *converges in one round.*

Proof. Let Π_C be the set of available paths between S and C discovered by Algorithm 1, and let $\Pi^*_C = \{P_C^1, \ldots, P_C^m\}$ be the optimal set of paths chosen by C for transmission, according to Theorems 1–3. If b_C^k represents the available rate of on requested path $P_C^k \in \Pi^*_C$, we have $b_C^k \leqslant \rho_u$, $\forall L_u \in P_C^k$. Since, by hypothesis, the chosen paths P_C^k do not contain any joint bottleneck link L_u, we have $\rho_u \geqslant \sum_{k:L_u \in P_C^k} b_C^k$, $\forall L_u \in P_C^k$ and $\forall P_C^k \in \Pi^*_C$. This means that any node N_i, upon the reception of reservation packets, *Resv*, can allocate the requested bandwidth on the outgoing links for all requested flows. Therefore, no flow is marked with $d^k = 0$, and the server S can compute the optimal allocation $\Pi^* = \Pi^*_C$, after one round of the protocol. □

Property 2. *Let the network graph that corresponds to the available resources at one stage of the algorithm be denoted $G' = \bigcup_{i:N_i \in V} \mathcal{N}'_i$ During each round,* Algorithm 1 *reserves in G' at least the end-to-end flow P_C^i between S and C which is affected by the smallest loss probability P_C^i.*

Proof. Let $P_C^i \in \Pi^*_C \backslash \Pi^*$ be the lowest loss probability path requested by C but not yet reserved by our algorithm. Observe that P_C^i is the lowest loss probability path in the residual graph G', and also in the local view $\mathcal{N}'_i$ observed by each node N_i. Hence, at every node N_i traversed by P_C^i, the flow P_C^i will have priority during the greedy reservation phase of Algorithm 1.

Indeed, from the path extension operation we have $b_C^i \leqslant \rho_u$, $\forall L_u \in P_C^i$. Hence P_C^i is successfully reserved at each

intermediate node N_i on the path. Finally, the flow P_C^i reaches S with the *Resv* packets with both flags $d^i = f^i = 1$, hence the server S integrates the flow to the set of successfully reserved paths: $\Pi^* = \Pi^* \bigcup P_C^i$. □

Property 3. Algorithm 1 *converges, and terminates in at most m rounds, where m is the number of allocated flows, which is moreover not larger than the total number of available distinct paths in G.*

Proof. This result is a direct consequence of Property 2. At each round, the algorithm reserves at least one flow, and the available rate of the links in the residual network decreases. Hence, in subsequent rounds of the algorithm, the client C will not be able to request an infinite number of flows. □

The previous properties show that given the available flow-equivalent network graph G, Algorithm 1 converges to the optimal path selection in a limited number of rounds, no more than the total number of available end-to-end paths between S and C. Moreover, in the case of disjoint network paths, our protocol manages to reserve the optimal set of flows needed for transmission in a single round. Finally, for the more general case, the algorithm secures at least one transmission flow from the optimal allocation.

We now concentrate on the second algorithm, and demonstrate that it converges in a single iteration. Moreover, we show that the solution offered by Algorithm 2 is actually identical to the optimal solution provided by Algorithm 1 if each network node has only one outgoing link.

Property 4. Algorithm 2 *converges after one round of path discovery and selection phases.*

Proof. Let Π_C be the set of available paths between S and C, as discovered in the path discovery phase of Algorithm 2, based on path extension Rule 2. Let further $\Pi^*_C = \{P_C^1, \ldots, P_C^m\}$ be the optimal set of paths chosen by C for transmission according to Theorems 1–3, based on the information received from the network nodes. Let finally b_C^k be the rate of the requested path $P_C^k \in \Pi^*_C$, with $b_C^k \leqslant \rho_u$, $\forall L_u \in P_C^k$. The greedy rate allocation in the path extension given by Rule 2 ensures that at any node N_i, and $\forall L_u \in O_i$, we have $\sum_{k:L_u \in P_C^k} b_C^k \leqslant \rho_u$. This means that any node N_i, upon reception of reservation packets, can allocate the bandwidth on the outgoing links for all requested flows. Therefore, no flow is marked with $d^k = 0$, and the server S can compute the optimal allocation $\Pi^* = \Pi^*_C$, after one round of the protocol. □

Property 5. Algorithm 2 *provides the same solution as* Algorithm 1 *if the outdegree of every intermediate node N_i is equal to 1.*

Proof. In this particular type of networks, we observe that the rate allocation operations during path extension in the path discovery phase becomes identical for both Algorithms 1 and 2. Since the rest of the algorithms is totally identical, they will provide the exact same solution, which is moreover optimal. □

Algorithm 2 offers a fast way of computing available end-to-end network paths between S and C. Since rate allocation decisions are taken independently at each intermediate node, the algorithm works for any type of loop-free DAGs.

5.2. Practical implementation

We discuss here the practical implementation of the proposed algorithms, and propose a few examples for deployment in real network scenarios. In large scale networks, monitoring end-to-end paths between any two given nodes becomes highly complex and costly. Nor active neither passive monitoring solutions scale well in terms of execution time, accuracy and complexity with a growing number of intermediate nodes and network segments [7]. Since full knowledge about network status cannot be achieved in large scale networks, distributed path computation solutions are certainly advisable. They additionally permit to release the computational burden of a single node/server, and distribute it among several intermediate nodes [16]. Networking protocols have been proposed to organize large scale random network graphs into DAGs [34], or sets of multiple end-to-end paths [32] and even to ensure special network properties like path disjointness and survivability [14].

We address the decentralized path computation and rate allocation problem, from the perspective of a media streaming application. The forwarding decisions are taken in order to maximize the quality of service of such specific applications, in particular to minimize the loss probability and aggregate enough transmission bandwidth. Our algorithms run at the application layer and present a low complexity in terms of message passing and execution time. In variable network scenarios, where the link parameters change slowly over time, our algorithms can be run periodically in order to adapt the rate allocation to a dynamic network topology. Observe that the fastest network parameter estimation algorithms offer good results on timescales of a few seconds [23], while the execution of our path computation algorithms takes one, or a few RTTs. Hence, running our algorithm periodically, on timescales equal to the network estimation intervals ensures efficient routing and rate allocation with the latest estimation about the network state. Finally, the control overhead can be limited to two packets on each link of the network, for each iteration of the distributed algorithms. For most typical scenarios, the overhead stays very low compared to the streaming rate. It typically depends on the periodicity chosen for the computation of the distributed rate allocation. We finally identify a few typical scenarios where effective rate allocation between multiple stream paths can bring interesting benefits in terms of media quality. The list is certainly not exhaustive, and it rather describes a few practical situations where the application of the algorithms proposed above is straightforward.

- Overlay network scenarios (e.g., structured peer-to-peer networks or Content Distribution Networks). The media information from a server is forwarded towards

the client by multiple edge servers or proxy, which belong to the same overlay network. The client consumes the aggregated media stream from multiple transmission flows employed by the application.

- Wireless network scenarios (e.g., WiFi networks). A wireless client can aggregate the media information transmitted on multiple wireless channels. Interference among transmission channels can be minimized by choosing non-overlapping wireless channels (e.g., there are eight non-overlapping channels according to the IEEE 802.11a standard specifications), and by optimizing the transmission schedule in the wireless network [31].
- Hybrid network scenarios (e.g., UMTS/GPRS/WiFi networks). A mobile client can simultaneously benefit from multiple wireless services in order to retrieve the media information from a server connected to the internet backbone.

Our framework for path selection and rate allocation can be applied to any of the above-mentioned scenarios in a straightforward manner. For the case of shared network resources in many-to-many setups, simple modifications to our algorithms can yield good resource allocations among clients, given an optimization metric. Consider Φ as the resource sharing policy implemented at an intermediate node *i*. Φ is designed according to the final optimization metric of the overall system, e.g., maximizing system quality [4]. Fairness and congestion control mechanisms on the end-to-end discovered paths can also be successfully applied [19], simple distributed resource sharing and packet prioritization schemes can be implemented based on the different importance of the simultaneous sessions [2]. Based on Φ, each node i can take an appropriate decision on how to allocate its resources, (namely the bandwidth of the outgoing links) among the concurrent applications, based on predefined utility functions for example. While the design of a truly fair distribution of resources between concurrent sessions is outside the scope of this paper, our generic framework permits to easily limit the bandwidth offered to a single session. The implementation of independent congestion control solutions becomes, therefore, straightforward.

6. Simulations

6.1. Simulation setup

We analyze the performance of our path computation algorithms in different network scenarios, and we compare them to simple heuristic-based rate allocation algorithms. Results are presented in terms of convergence time, and video quality performance. We first study the average behavior of the algorithms in random network graphs, and we eventually discuss in details a specific scenario, implemented in ns2 [8] in the presence of cross traffic.

In all simulations, the test image sequence is built by concatenation of the *foreman* sequence, in CIF format, in order to produce a 1500-frame video stream, encoded in H.264 format at 30 frames per second (equivalent to 50 s of video). The encoded bitstream is packetized into a sequence of network packets, where each packet contains information related to at most one video frame. The size of the packets is limited by the size of the maximum transmission unit (MTU) on the underlying network. The packets are sent through the network on the chosen paths, in a FIFO order, following a simple scheduling algorithm [11]. The video decoder finally implements a simple frame repetition error concealment strategy in case of packet loss. A video packet is correctly decoded at the client, unless it is lost during transmission due to the errors on the network links, or unless it arrives at the client past its decoding deadline. We consider typical video-on-demand (*VoD*) streaming scenarios, where the admissible playback delay is large enough, i.e., larger than the time needed to transmit the biggest packet on the lowest bandwidth path.

6.2. Random network graphs

We generate two types of network topologies: (i) typical *Wireless* network graphs, with low bandwidth and high error probability for the network links and (ii) *Hybrid* network scenarios, where the server is connected to the wired infrastructure (high rate, low loss probability), and the client can access the internet via multiple wireless links, that have a reduced bandwidth, and an increased loss probability. For both scenarios, we generate 500 random graphs. Any two nodes in the graph are directly connected with a probability γ. The parameters for each link are randomly chosen according to a normal distribution, in the interval $[R_{\min}, R_{\max}]$ for the bandwidth, and, respectively, $[p_{\min}, p_{\max}]$ for the loss probability. The parameters for representative wired and wireless links are presented in Table 1.

We limit the size of the generated graphs to 15 nodes. In this case, the number of available paths between the server and the client is already much larger than the number of paths actually chosen for the media transmission (6.89 and 6.73 available paths in average for the *Hybrid* and *Wireless* case, respectively). The results below can be easily extended to larger networks, where the graph processing algorithms, however, rapidly become the computational bottleneck. Note that these algorithms are independent of path selection and distributed rate allocation strategies studied in this paper.

First, we analyze the number of rounds in which Algorithm 1 converges to the optimal rate allocation given by a centralized algorithm, as proposed in Ref. [10]. The results for both network scenarios are presented in Fig. 3. We observe that the large majority of the cases require less than three iterations in order to reach the optimal rate allocation. This shows that our algorithm performs very fast and needs only a very small number of control messages to converge to the optimal rate allocation.

Next, we propose to examine in Fig. 4 the convergence of Algorithm 1, computed in terms of video distortion, as compared to the quality of the stream achieved with the optimal rate allocation. We observe that the distortion due to Algorithm 1 rapidly decreases, and that the partial

Table 1
Parameters for random graph generation

Parameter	Wired links	Wireless links
Connectivity probability γ	0.4	0.6
R_{min}	10^6 bps	10^5 bps
R_{max}	3×10^6 bps	7×10^5 bps
p_{min}	10^{-4}	10^{-3}
p_{max}	5×10^{-3}	4×10^{-2}

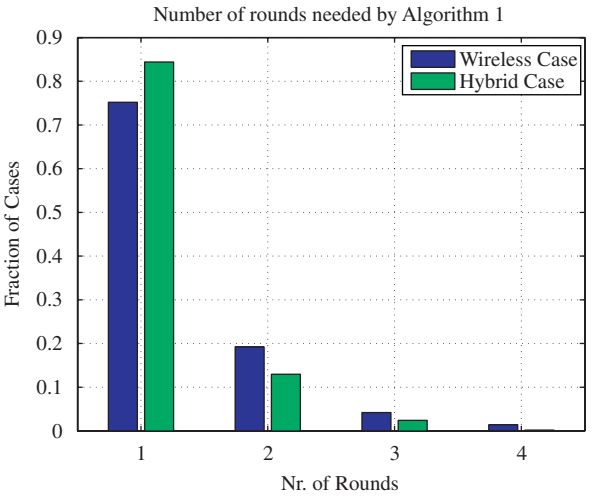

Fig. 3. Probability density function of the number of rounds necessary for the iterative rate allocation to converge to the optimal solution of Algorithm 1.

solutions are very close to the optimal one, even after the first round of the iterative rate allocation strategy. It clearly illustrates that the proposed distributed algorithm converges very fast to the optimal solution, and that the most critical paths in terms of video quality are already allocated by the very initial rounds of the distributed solution.

In both Figs. 3 and 4, we can observe that Algorithm 1 performs better in the *Hybrid* network scenario, than in the *Wireless* case. This is due to the fact that this network scenario has in average less bottleneck links. Please observe that in this simulated scenario, the bottleneck links are usually the wireless links, since the rates of the wired links are much higher. Therefore, Algorithm 1 is expected to converge faster to the optimal solution in the *Hybrid* scenario, where path are less likely to share bottleneck links. This is in accordance with the properties of this algorithm presented in the previous section.

Then, we analyze the performance of the proposed algorithm, in terms of video quality obtained with the rate allocation solution. We compare the results obtained with Algorithm 1, to the ones obtained by a simpler distributed heuristic that forwards the incoming network flow at each intermediate node on the best outgoing link in terms of loss probability (e.g., single best-path streaming). We compute the distribution of the penalty in quality suffered by the heuristic scenario, for 500 different network graphs. Results are represented in Fig. 5. They illustrate

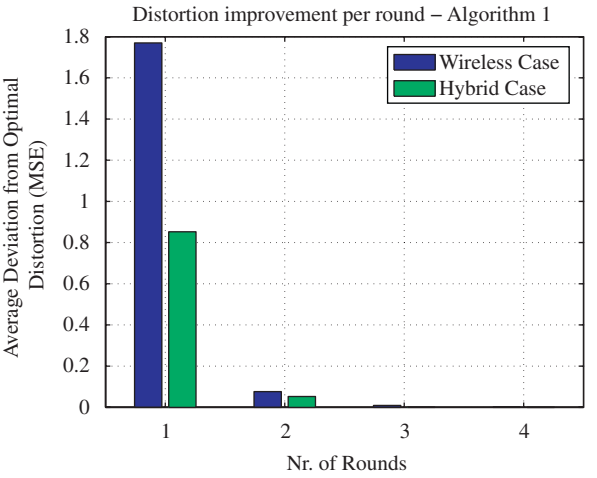

Fig. 4. Convergence of Algorithm 1, measured in terms of video distortion (MSE), as compared to the optimal solution.

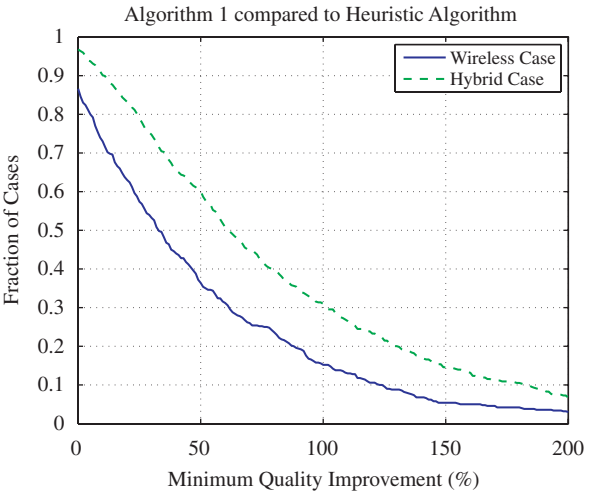

Fig. 5. Improvement in quality offered by Algorithm 1 vs. a heuristic rate allocation algorithm.

the probability for the improvement in quality to be above a predefined threshold x. We observe that, for both network scenarios, our algorithm obtains significantly better results in more than 80% of the cases. This motivates the extra control overhead introduced by Algorithm 1, which is needed to reach the optimal rate allocation. A similar behavior is shown in Fig. 6, where we observe that Algorithm 2 also performs much better than the single best path strategy, in a large fraction of the cases considered, and for both network scenarios.

Algorithms 1 and 2 are compared in Figs. 7 and 8. Fig. 7 represents the difference in quality incurred by Algorithm 2, with respect to the optimal allocation offered by Algorithm 1. A similar representation is proposed in Fig. 8, where the quality provided by Algorithm 1 is computed based on the rate allocation obtained after the first round of the iterative algorithm, as opposed to the optimal allocation that is used in Fig. 7. From both figures,

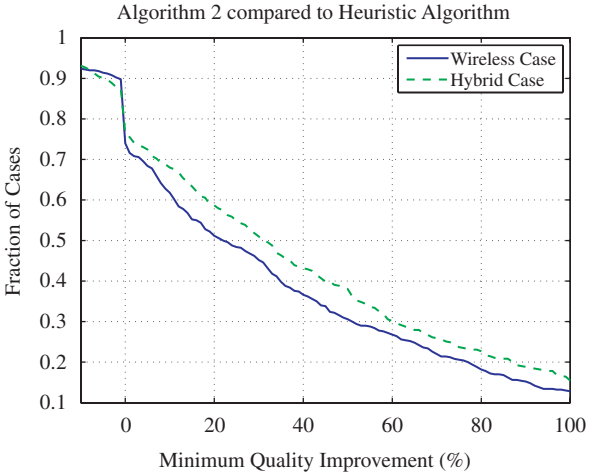

Fig. 6. Improvement in quality offered by Algorithm 2 vs. a heuristic rate allocation algorithm.

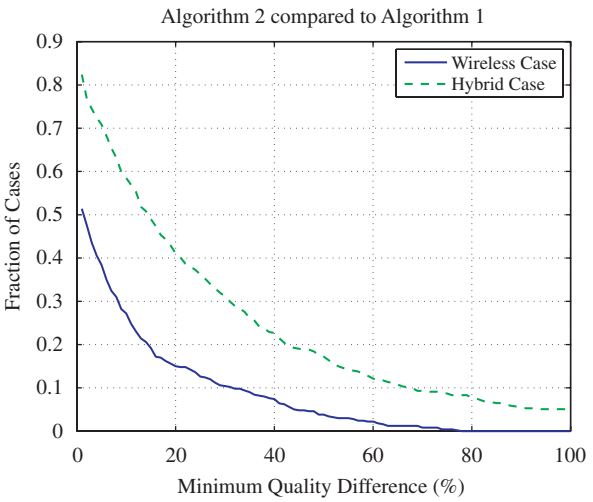

Fig. 7. Improvement in quality for Algorithm 1 vs. Algorithm 2.

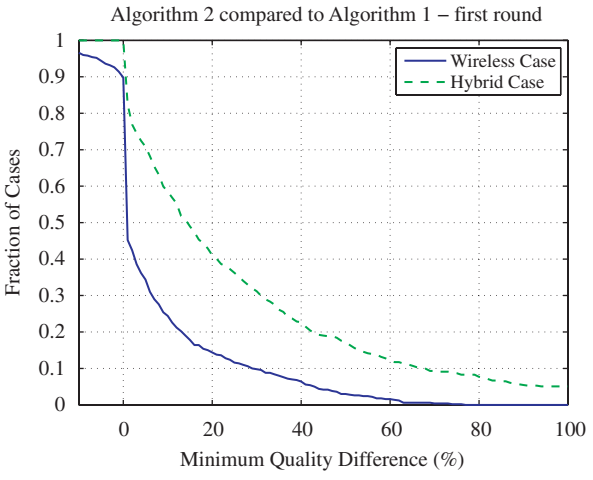

Fig. 8. Improvement in quality for Algorithm 1 limited to one iteration only, vs. Algorithm 2.

we see that, for the *Wireless* scenario, the performance of the greedy scheme is equal to the optimal solution in more than 50% of the cases. Algorithm 2 is even better, when compared to the execution of the optimal algorithm after the first round (80% of the cases providing similar results). This is due to the very small number of paths chosen for transmission, and to the fact that link parameters in the *Wireless* scenario are quite homogeneous. In the pathological case where all network links would have the same parameters, the performance of the two algorithms would be identical. However, in the *Hybrid* network scenario, we observe that the greedy algorithm offers bad results in a significant number of cases, since quality attains only 50% of the optimal solution in almost 20% of the cases. This is mainly due to the heterogeneity of the network links parameters in hybrid scenarios.

Finally, we compare Algorithms 1 and 2 in terms of number of paths chosen for the streaming application. The results for the *Wireless* and *Hybrid* network scenarios are presented in Figs. 9 and 10, respectively. We observe that in general Algorithm 2 uses a smaller number of flows for transmission. This can be explained by the greedy allocation of paths, when Rule 2 is used for path extension. Similar results can be observed when the average streaming rate is computed for the solutions provided by both algorithms, for each type of networks. Table 2 shows that Algorithm 2 generally results in a smaller transmission rate. However, the performance in terms of received video quality is very close to the optimal one, since the paths with the lowest loss probability are prioritized in both algorithms. In addition, the particular network setup used in the simulation allows for average streaming rates that already offer a good encoding quality, where the rate-distortion gradient is not very large.

Overall, the previous results show that Algorithm 1 represents a fast path computation solution in most types of networks that present a low number of bottleneck links. On the other side, Algorithm 2 offers a viable, lower complexity alternative for very large network scenarios

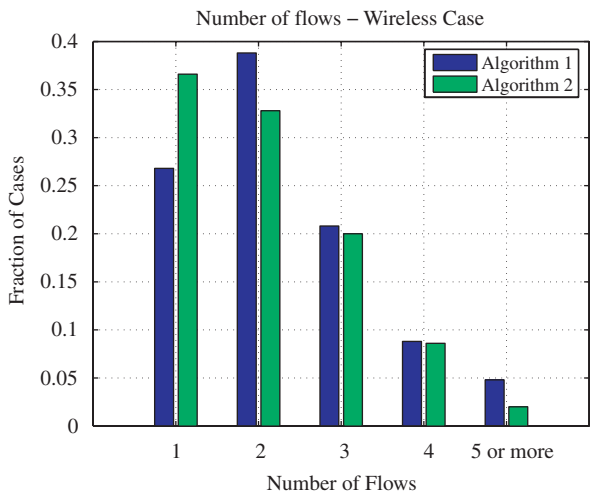

Fig. 9. Probability density function of the number of paths used by Algorithms 1 and 2 in the *Wireless* Network Case.

with homogeneous link parameters, where convergence time is an issue (e.g., in networks characterized by quickly varying parameters).

6.3. Sample network scenario

We now compare the performance of the two path computation algorithms presented in this paper, in a specific network scenario that represents a practical case study in wired networks. We transmit the *foreman* sequence, encoded at 375 and 550 kbps over the network presented in Fig. 11(a). The network scenario is reproduced in the ns2 simulator, and the algorithms presented here are implemented as extensions to the simulator. On each of the network paths from the server to the client, we simulate 10 background flows. These flows are generated according to an on/off source model with exponential distribution of staying time, and average rates between 100 and 300 kbps. The instantaneous rate available to the streaming application is considered to be the difference between the total link bandwidth, and the instantaneous rate of the aggregated background traffic. We generate two network cases, one with low average link rates and high transmission error probability (i.e., end-to-end loss probability higher than 6%), and a second case with higher average link rates and average transmission error probability (i.e., end-to-end loss probability of about 3%). The average bandwidth, and loss probabilities are presented in Table 3, for the two cases under consideration. The network MTU is set to 1000 bytes worth of video data. Finally, we also consider cases where the video stream is sent along with forward error protection. Overhead packets are sent in addition to the video packets for packet loss recovery. FEC blocks of 20 packets are formed by adding two redundant packets for each set of 18 video packets in the first network case. In the second case, one FEC packet is added to each group of 19 video packets. Therefore, all video packets can be recovered if at least 18, respectively 19 packets are correctly received in a block of 20 packets. Note that in this specific scenario, both strategies result in an overall streaming rate that is smaller than the average aggregated bandwidth available on the network. Distortion is mostly caused by packet losses, or late arrival due to bandwidth fluctuations.

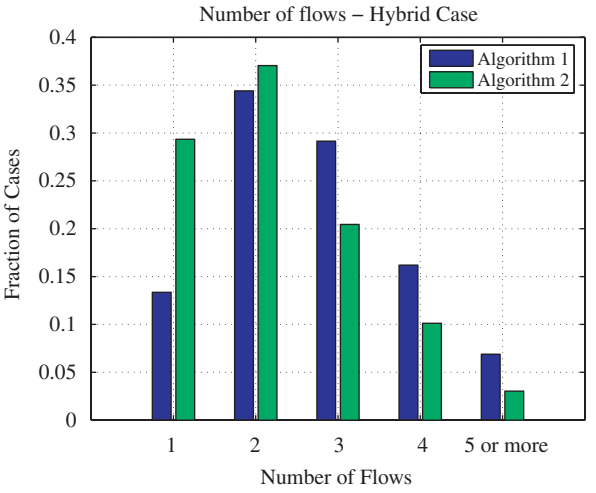

Fig. 10. Probability density function of the number of paths used by Algorithms 1 and 2 in the *Hybrid* Network Case.

Table 2
Average transmission rates chosen by Algorithms 1 and 2

	Wireless (kpbs)	Hybrid (kpbs)
Algorithm 1	625	957
Algorithm 2	527	681

Fig. 11(b and c) first show the path selection provided by Algorithms 1 and 2, respectively. Both network cases result in the same allocation and the application packets and the control messages of our algorithms share the same network links. Simulations are then run according to these path allocations, and each simulation point is averaged over 10 simulation runs. Figs. 12 and 13 present the performance of Algorithms 1 and 2 as a function of the playback delay imposed by the client, respectively in absence or presence of FEC protection. Recall that the server performs a simple round-robin packet scheduling strategy, for a given set of streaming path. Hence, the playback delay influences the scheduling performance, and larger playback delays permit to pay smaller penalty due to the scheduler choices. The video distortion values incorporate the source distortion due to the low encoding rate of the sequence, along with the loss distortion due to packet transmission loss, and late arrivals at the client. We observe that even if the choice of transmission paths differs between the two algorithms, the performance is similar, since the end-to-end paths are disjoint, and quite homogeneous in the network case under study. It can be

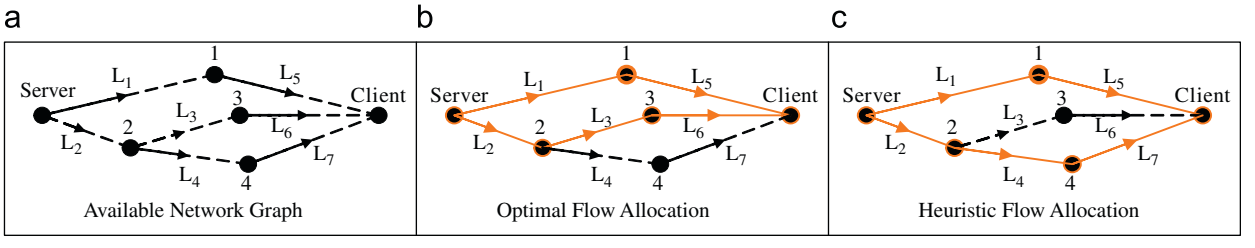

Fig. 11. Network scenario: (a) available network graph, (b) flow allocation chosen by Algorithm 1 and (c) flow allocation chosen by Algorithm 2.

Table 3
Parameter values for the network links in Fig. 11

	L_1	L_2	L_3	L_4	L_5	L_6	L_7
Case 1: loss (%)	2.0	1.0	2.0	1.5	1.5	0.5	2.5
Case 1: rate (kbps)	325	225	225	225	325	225	225
Case 2: loss (%)	1.5	1.0	1.0	0.75	1.0	0.5	1.5
Case 2: rate (kbps)	450	300	300	300	450	300	300

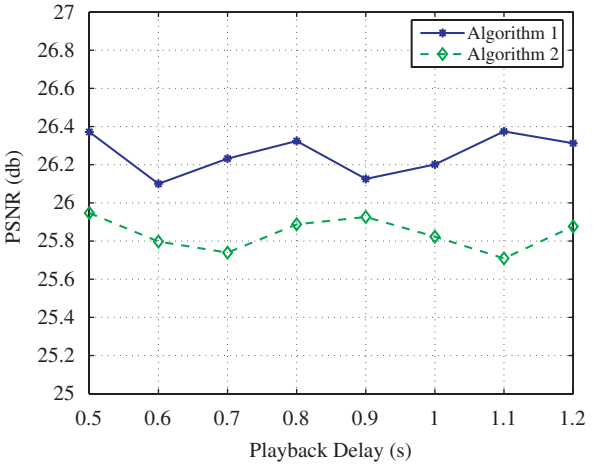

Fig. 12. Performance evaluation of Algorithms 1 and 2 as a function of playback delay (Network Case 1, no FEC).

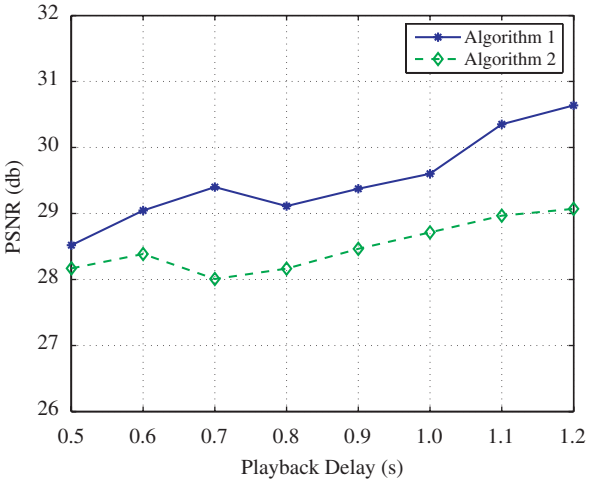

Fig. 13. Performance evaluation of Algorithms 1 and 2 as a function of playback delay (Network Case 1, with FEC).

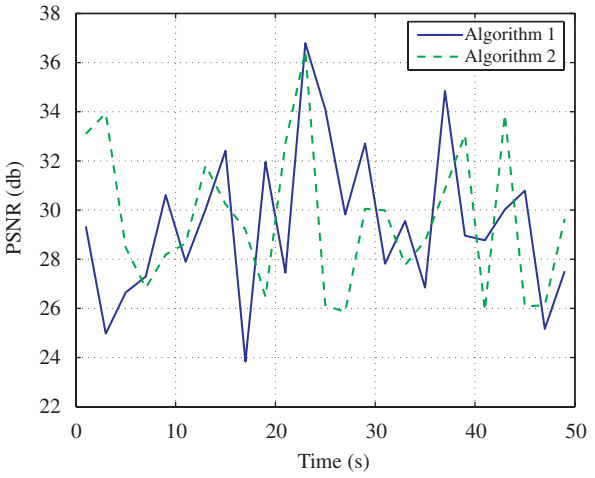

Fig. 14. Temporal evolution of the video quality (Network Case 2, no FEC).

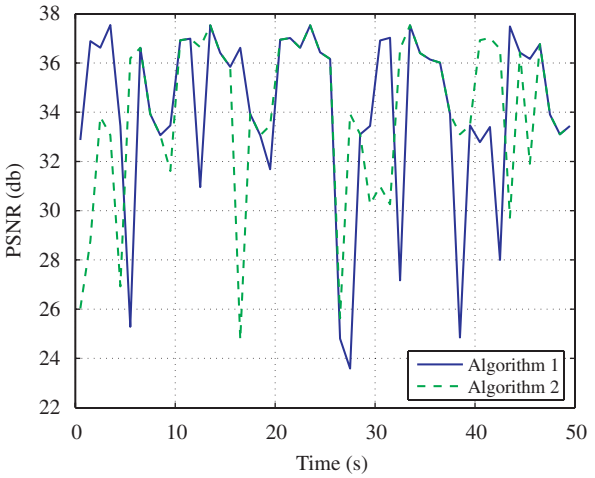

Fig. 15. Temporal evolution of the video quality (Network Case 2, with FEC).

noted that the influence of the playback delay is similar for both schemes. At the same time, it can be observed that using even a minimum error protection strategy unsurprisingly improves the final results. The impact of the playback delay on the distortion measure compared to transmission errors is also more visible in this case, as the latter are mostly compensated for by the FEC scheme. Very similar results can be observed for the second network case with the 500 kbps video bitstream, but they are omitted here due to space constraints.

Finally, we pick one of the simulation runs for each algorithm, and analyze the temporal evolution of the

quality. The average video quality is measured at the receiver for each group of 30 pictures, with or without FEC, respectively. Results are presented in Figs. 14 and 15 for the second network case, where the playback delay imposed by the client is set to one second. It can be seen that both rate allocation algorithms again perform similarly in the presence of packet loss and cross traffic. The quality fluctuations are mostly due to packet loss, and basic FEC protection already helps to improve the decoded quality. It confirms the results presented above and positions both algorithms as interesting solutions for distributed and media-specific rate allocation in multi-path networks.

7. Conclusions

This paper has addressed the problem of decentralized path computation for multimedia streaming applications in large scale networks. When end-to-end monitoring at the media server becomes intractable and expensive, distributed mechanisms need to be derived in order to optimize the streaming process in terms of media quality. We present two such mechanisms for path computation that differ in the construction of available paths between the streaming server and the client on a node-by-node basis. The first algorithm provides a comprehensive view of the set of end-to-end paths, which leads to optimal rate allocation, at the price of a small convergence time. The second algorithm only offers partial information about the available paths, which results in a lower complexity solution. However, thanks to a greedy allocation that favors the most reliable paths, the performance of the second algorithm stays close to the optimal performance in most of the cases.

In both algorithms, each node is responsible for a rate allocation decision for all incoming flows, on the outgoing links. Hence, the set of available transmission paths between the server and the client is created only from the original local network views at each individual intermediate node. It permits to avoid the assumption of full network knowledge at any single node in the network and eliminates the need for expensive path monitoring mechanisms. Both solutions, therefore, represent interesting alternatives for media-specific path selection in medium to large scale overlay networks. In particular, extensive simulations demonstrate that the optimal algorithm converges very fast, especially in networks that present a small number of bottleneck links. At the same time, the greedy algorithm represents a viable and low complexity solution in very large network scenarios with homogeneous link parameters and stringent limitations on the convergence time of the algorithm.

References

[1] A.C. Begen, Y. Altunbasak, O. Ergun, M.H. Ammar, Multi-path selection for multiple description video streaming over overlay networks, Signal Process.: Image Commun. 20 (2005) 39–60.

[2] J. Chakareski, P. Frossard, Rate-distortion optimized distributed packet scheduling of multiple video streams over shared communication resources, IEEE Trans. Multimedia 8 (2) (April 2006) 207–218.

[3] Y. Cui, Y. Xue, K. Nahrstedt, Optimal resource allocation in overlay multicast, J. IEEE Trans. Parallel Distributed Syst. 17 (8) (August 2006) 808–823.

[4] M. Dai, D. Loguinov, H.M. Radha, Rate-distortion analysis and quality control in scalable internet streaming, IEEE Trans. Multimedia 8 (6) (December 2006) 1135–1146.

[5] P. Frossard, O. Verscheure, Joint source/fec rate selection for quality-optimal mpeg-2 video delivery, IEEE Trans. Image Process. 10 (12) (Dec 2001) 1815–1825.

[6] L. Golubchik, J. Lui, T. Tung, A. Chow, W. Lee, Multi-path continuous media streaming: what are the benefits?, ACM Journal of Performance Evaluation 49 (1–4) (September 2002) 429–449.

[7] http://www.icir.org/models/tools.html.

[8] http://www.isi.edu/nsnam/ns/.

[9] D. Jurca, Adaptive Media Streaming over Multipath Networks, Ph.D. dissertation, EPFL, October 2007.

[10] D. Jurca, P. Frossard, Media-specific rate allocation in multipath networks, IEEE Trans. Multimedia 9 (October 2007) 1227–1240.

[11] D. Jurca, P. Frossard, Video packet selection and scheduling for multipath streaming, IEEE Trans. Multimedia 9 (April 2007) 629–641.

[12] D. Jurca, S. Petrovic, P. Frossard, Media aware routing in large scale networks with overlay, in: Proceedings of the IEEE ICME, July 2005.

[13] J. Kim, M. Mersereau, Y Altunbasak, Distributed video streaming using unbalanced multiple description coding and forward error correction, in: Proceedings of the IEEE Globecom, 2003.

[14] M. Kurant, P. Thiran, Survivable routing of mesh topologies in ip-over-wdm networks by recursive graph contraction, To appear in JSAC special issue on Traffic Engineering for Multi-Layer Networks, 2007.

[15] Y. Li, S. Mao, S.S. Panwar, The case for multimedia transport over wireless ad hoc networks, in: Proceedings of IEEE/ACM BroadNets, October 2004.

[16] N. Lynch, Distributed Algorithms, Morgan Kaufmann Publishers Inc., San Mateo, CA, 1996.

[17] S. Mao, S. Lin, S.S. Panwar, Y. Wang, E. Celebi, Video transport over ad hoc networks: Multistream coding with multipath transport, IEEE J. Sel. Areas Commun. 21 (10) (December 2003) 1721–1737.

[18] D.L. Mills, Network time protocol implementation, specification and analysis, Network Working Report RFC 1305, Tech. Rep., March 1992.

[19] J. Mo, J. Walrand, Fair end-to-end window-based congestion control, IEEE/ACM Trans. Networking 8 (5) (2000) 556–567.

[20] T. Murakami, M. Bandai, I. Sasase, Split multi-path routing protocol with load balancing policy (smr-lb) to improve tcp performance in mobile ad-hoc networks, J. IEICE Trans. Commun. E89-B (5) (2006) 1517–1525.

[21] T. Nguyen, A. Zakhor, Path diversity with forward error correction (pdf) system for packet switched networks, in: Proceedings of IEEE INFOCOM, vol. 3, 2003, pp. 663–672.

[22] T. Nguyen, A. Zakhor, Multiple sender distributed video streaming, IEEE Trans. Multimedia 6 (2) (April 2004) 315–326.

[23] V. Ribeiro, R. Riedi, R. Baraniuk, J. Navratil, L. Cottrell, pathChirp: efficient available bandwidth estimation for network paths, in: Proceedings of the Passive and Active Measurement Workshop, April 2003.

[24] R. Rodrigues, B. Liskov, L. Shrira, The design of a robust peer-to-peer system, in: Proceedings of the SIGOPS European Workshop, 2002.

[25] S. Sarkar, L. Tassiulas, A framework for routing and congestion control for multicast information flows, IEEE Trans. Inf. Theory 48 (10) (October 2002) 2690–2708.

[26] S. Savage, A. Collins, E. Hoffman, The end-to-end effects of internet path selection, in: Proceedings of ACM SIGCOMM, 1999.

[27] E. Setton, B. Girod, Congestion–distortion optimized scheduling of video over a bottleneck link, in: Proceedings of the IEEE MMSP, 2004.

[28] J.L. Sobrinho, Algebra and algorithms for qos path computation and hop-by-hop routing in the internet, J. IEEE/ACM Trans. Networking (TON) 10 (4) (August 2002) 541–550.

[29] K. Stuhlmuller, N. Farber, M. Link, B. Girod, Analysis of video transmission over lossy channels, IEEE J. Sel. Areas Commun. 18 (6) (June 2000) 1012–1032.

[30] S. Tao, R. Guerin, Application-specific path switching: a case study for streaming video, in: Proceedings of the ACM Multimedia, October 2004, pp. 136–143.

[31] P. von Rickenbach, S. Schmid, R. Wattenhofer, A. Zollinger, A robust interference model for wireless ad-hoc networks, in: Proceedings of the IEEE WMAN'05, 2005.

[32] S. Vutukury, J. J. Garcia-Luna-Aceves, Mdva: A distance-vector multipath routing protocol, in: Proceedings of IEEE INFOCOM, vol. 1, 2001, pp. 557–564.
[33] W. Wei, A. Zakhor, Robust multipath source routing protocol (rmpsr) for video communication over wireless ad-hoc networks, in: Proceedings of the IEEE International Conference on Multimedia and Expo' (ICME2004), 2004.
[34] W.T. Zaumen, J.J. Garcia-Luna-Aceves, Loop-free multipath routing using generalized diffusing computations, in: Proceedings of the INFOCOM (3), 1998, pp. 1408–1417.
[35] X. Zhu, B. Girod, Distributed rate allocation for multi-stream video transmission over ad-hoc networks, in: Proceedings of the IEEE ICIP, 2005.