SCHOOL OF ENGINEERING - STI
SIGNAL PROCESSING INSTITUTE
Dan Jurca and Pascal Frossard

CH-1015 LAUSANNE

Telephone: +4121 6932601
Telefax: +4121 6937600
e-mail: {dan.jurca,pascal.frossard}@epfl.ch

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

DISTRIBUTED MEDIA RATE ALLOCATION IN MULTIPATH NETWORKS

Dan Jurca and Pascal Frossard

Swiss Federal Institute of Technology Lausanne (EPFL)

Signal Processing Institute Technical Report

TR-ITS-2006.009

September 14th, 2006

Part of this work has been submitted to IEEE TMM.

This work has been supported by the Swiss NSF, under grant PP002-68737.

Distributed Media Rate Allocation in Multipath Networks

Dan Jurca and Pascal Frossard
Ecole Polytechnique Fédérale de Lausanne (EPFL)
Signal Processing Institute
CH-1015 Lausanne, Switzerland
Email: {dan.jurca,pascal.frossard}epfl.ch

Abstract

The paper addresses the media-specific rate allocation problem in multipath networks. The streaming rate on each path is determined such that the end-to-end media distortion is minimized, when the receiving client aggregates packets received via multiple network channels. As it is difficult for the media server to have the full knowledge about the network status, we propose a distributed path selection and rate allocation algorithm. The network nodes participate to the optimization strategy, based on their local view of the network status. This eliminates the need for end-to-end network monitoring, and allows for the deployment of large scale rate allocation solutions. We design an optimal rate allocation algorithm, where the media client iteratively updates the best set of streaming paths. According to this rate allocation, each intermediate nodes then forwards incoming media flows on the outgoing paths, in a distributed manner. The proposed algorithm is shown to quickly converge to the optimal rate allocation solution, and hence to lead to stable rate allocation solutions. We also propose a greedy distributed 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, and therefore offer an interesting solution for media-specific rate allocation over large scale multi-path networks.

I. Introduction

As the internet is far from providing any widely deployed guarantee of service solution, 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. Lately, multipath streaming emerged as an effective solution to overcome some of the lossy internet path limitations [1], [2]. It allows for 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 through various network paths.

This paper addresses the problem of distributed media-specific rate allocation for streaming applications in multipath networks. We build on our prior work [3] that provides a general framework for the analysis of joint path and rate allocation in multipath streaming, driven by media-specific quality metrics. It considers a network model composed of multiple flows, and a streaming server that can adapt the media source rate to the transmission conditions (by scalable coding, or transcoding, for example). Given the knowledge of the network parameters, the server selects an optimal set of transmission paths, along with their respective transmission rate. However, such a strategy requires end-to-end monitoring, and the knowledge of the complete network status at the server, which clearly limits the implementation of such algorithms to small-scale network scenarios. Therefore, we propose in this paper a distributed solution, where intermediate network nodes capable of handling application-level information, participate to the path selection and rate allocation algorithm, based on their local view of the network.

The joint path selection and rate allocation performs iteratively, until all intermediate nodes converge to a (unique) optimal solution. Initially, the intermediate network nodes together report the resources available to the streaming session. Based on this information, the client determines the best path selection and rate allocation, and issues 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. Such a distributed strategy allows to relax the assumption of full network status knowledge at the server, and to eliminate the need of end-to-end network monitoring systems. We design an optimal path selection algorithm that quickly converges to the optimal 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 outperforms other solutions in terms of media quality metrics, and provide effective solutions to streaming rate allocation in medium to large scale multipath networks.

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

The rest of this paper is organized as follows. Section II discusses the related work and motivates the need for distributed rate allocation solutions. Section III describes in detail the streaming scenario considered in this paper, and presents the rate allocation optimization problem. We present our distributed solutions in Section IV and we analyze the characteristics of the proposed algorithms in Section V. Extensive simulation results are finally presented in Section VI, for numerous network topologies, and for a practical scenario that is analyzed in details.

II. 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. Previous works discuss similar problems, but always consider the path selection problem from a network point of view only. Numerous routing algorithms have been proposed, to optimize a given network QoS metric [4], to improve the performance of TCP over wireless Ad-Hoc networks [5], to discover multiple available network paths to one source [6], or to optimize the network resource allocation in overlay multicasts [7], [8]. In addition, the authors of [9] 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 media streaming application. In 30-80% of the cases, the best paths found by classic routing algorithms are suboptimal from a media perspective [10].

In the same time, several works have investigated the problem of multipath streaming, as a way to increase the media 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 between a server and a client. More specifically, the multipath problem is specifically addressed in the case of media streaming in [11], 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. Multi-stream coding, combined with multipath transmission, has been presented in [12] as a solution to fight against network errors in an ad-hoc network environment. In the same time, the authors of [13] analyze a multiple path streaming scenario for the transmission of a video sequences encoded in multiple descriptions. Other works in distributed video streaming [14]–[16] 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 them specifically targets the optimal choice of the streaming paths, and the relative rate allocation problem.

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. The work presented in [17] addresses a similar 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 addition to considering multipath strategies that have been shown to improve streaming performances, our work innovates by proposing a distributed solution for path computation. It alleviates the need for expensive end-to-end path monitoring systems, and hence can be deployed in large scale network scenarios. Our streaming framework is quite generic, and applicable to any streaming system that obeys an additive rule for the aggregated transmitted rate and loss process. 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.

III. The multipath rate allocation problem

A. Network and Video Model

We consider that the media streaming application is deployed on a large scale network, modeled as a fully connected directed acyclic graph $G(V, E)$, between the streaming server S and the client C (Figure 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 to, and from node N_i. Each link $L_u \in E$ has two associated positive metrics:

- the available bandwidth $\rho_u > 0$ expressed in some appropriate unit (e.g., kbps), and,
- the average packet loss probability $p_u \in [0, 1)$, assumed to be independent of the streaming rate.

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

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

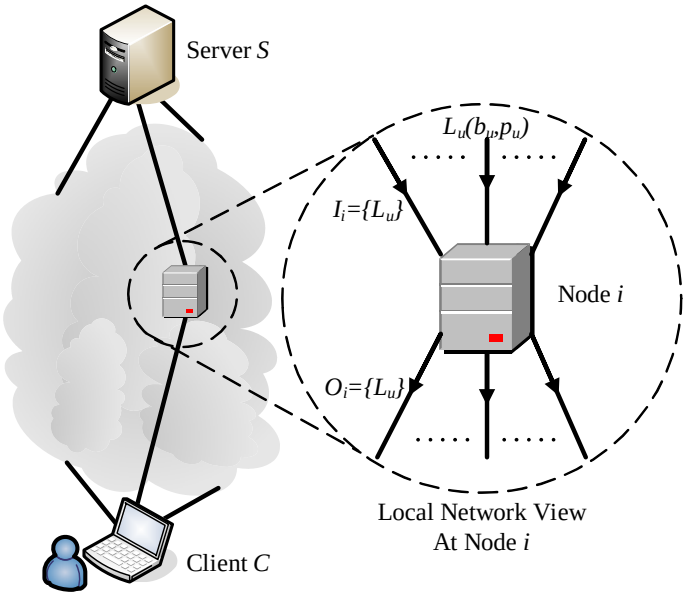

Fig. 1. Multipath Network Scenario and Network View at Node N_i.

The video quality is then assumed to be dependent on the actual streaming rate and 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. 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 ϵ of 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 ϵ corresponds to the actual video packet loss ratio when video frames are encapsulated into distinct network packets. The end-to-end distortion can thus be written as:

$$D = D_S + D_L = \alpha \cdot R^{\xi} + \beta \cdot \epsilon, \tag{1}$$

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 :

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

In this paper, we assume that the streaming server can tune the media source rate to the transmission conditions (by scalable coding, or transcoding, for example). In the same time, when the loss processes on different paths are independent, the overall loss probability becomes :

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

The 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. Note that under the given network assumptions, the video distortion metric is insensitive to the actual link error model, and is only influenced by the average loss probability on the given network segment. A validation of this model through video experiments can be found in [3].

The remainder of this section presents the optimization problem, whose aim is to find the optimal flow rate allocation in order to maximize the received media quality at the client. In the same time, we provide an overview of the solution in the case where the available end-to-end network paths are known in advance by the server. We then present the distributed optimization problem that is solved in the rest of the paper. The assumption on full network status knowledge at a given node can therefore be released, and the need of end-to-end monitoring mechanisms eliminated.

B. 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 previous work [3], we have derived the analytical rules that allow to derive the optimal solution to the joint path selection and rate allocation problem, 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 network graph. Interested readers are referred to [3] for ample discussions and proofs of these theorems.

Theorem 1 (On-Off Flows): Given a network graph G with independent flows $\mathcal{F}_C^i$ 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, $\forall i : 1 \leq i \leq n$.

Theorem 2 (Parameter Decoupling): Given a network graph G with independent flows $\mathcal{F}_C^i$ 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 $\Phi^* = [b_C^1, b_C^2, ..., b_C^i, 0, 0, ...0]$.

Theorem 3 (Bottleneck Bandwidth Sharing): Let L_u be a bottleneck link for the set of paths $\mathcal{B}_u = \{P_C^k\}$ 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 to 3 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 process. In 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 generic network scenarios. We now relax the assumption of full network knowledge at the streaming server, and present distributed mechanisms for computing the available end-to-end paths on the network graph. We build on the previous theorems to eventually compute the optimal rate allocation on these paths.

C. 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 (e.g., by scalable encoding, transcoding or packet filtering), and that packet loss rate is independent of the streaming rate, the server S 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. The optimization problem can be formulated as follows:

Distributed Multimedia Rate Allocation Problem (DMMR): Given the 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 $\mathcal{N}_i$, $\forall N_i \in V$ and given the video sequence characteristics ($\Gamma = (\alpha, \beta, \xi)$), find the complete set of end-to-end paths P_C^i, $1 \leq i \leq n$ and the optimal rate allocation $\vec{R^*} = [r_C^1, ...r_C^n]^*$ that minimizes the distortion metric D:

$$\vec{R^*} = \arg \min_{\vec{R}} (\alpha \cdot R^{\xi} + \beta \cdot \epsilon) \ , \tag{2}$$

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

IV. Distributed Rate Allocation

A. 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 details 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 has an estimate of the bandwidth ρ_u and loss probability p_u on 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

[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 form self-congestion.

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 define $\rightarrow$ as the path extension operator, which 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, 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 $\Omega_i = \{P_j^l = P_i^k \rightarrow L_u \mid P_i^k \in \Pi_i, L_u \in O_i\}$. The subset of the extended paths that borrow the particular outgoing link L_u is written as $\Omega_i^u = \{P_j^l = P_i^k \rightarrow L_u \mid P_i^k \in \Pi_i\}$. 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 the number of outgoing links [18]. 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 increasing values of loss probability, the set of extended paths at node N_i can be written as $\Gamma_i = \{P_j^l = P_i^k \rightarrow L_u \mid \sum_{\mu=1}^{u} \rho_\mu > \sum_{\nu=1}^{k-1} b_i^\nu \text{and} \sum_{\mu=1}^{u-1} \rho_\mu < \sum_{\nu=1}^{k} b_i^\nu\}$. 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 Rule 2, we now describe the rate allocation strategy.

B. 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 client, and then by reserving paths according to the optimal allocation computed by the client. It proceeds in two phases, the path discovery, and the path reservation phases, respectively. To this aim, control messages are exchanged between the server S and the client C via forwarding by the intermediate nodes. We assume the existence of a bidirectional control channel between any two nodes in G that are connected by a network segment L_u. In order to derive exact bounds on the performance of our algorithms, we assume that the control channel is reliable, and that nodes are synchronized (i.e., there is a bounded time interval in which all nodes receive all dedicated control packets). Note that these assumptions are not crucial to the design of the proposed algorithms. For example, loose node synchronization is assumed in most works addressing decentralized systems [19] in order to obtain concrete bounds on protocol performance. This could be easily achieved by employing separate synchronization protocols [20].

The server sends on all outgoing links path discovery messages, $Path^u$, which are forwarded by the intermediate nodes 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 eventual information related to previously successfully reserved flows. The node then extends the path according to Rule 1 or Rule 2, and forwards path discovery message $Path^u$ that basically contains information about the paths that borrow links L_u. Depending on the path extension strategy, the client will eventually receive information about all possible paths, or only a subset of them that are computed in a greedy manner, based on decreasing reliability.

Upon reception of path discovery messages, the client C computes the optimal path selection Π_C^* using the Theorems 1 to 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, since they state that paths should be either used at their full bandwidth, or simply dropped. The client then initiates path reservation messages, $Resv^u$, which are forwarded by the network nodes to the server, on the backward control channel associated with link[3] L_u. A path reservation message $Resv^u$ contains information about the path(s) that should be reserved on link L_u for the streaming session (e.g., requested rate b_C^k, end-to-end loss probability p_C^k and flags f^k and d^k, 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_C^k. It eventually marks the flows that cannot be reserved at the requested rate b_C^i, by setting the flag $d^k = 0$. Once a valid subset of paths $\Pi^* \subseteq \Pi_C^*$ is

[3]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.

Algorithm 1 Distributed Path Selection and Rate Allocation Algorithms

server S:	node N_i:
upon receive $Resv^u$, $\forall L_u \in O_S$:	upon receive $Resv^u$, $\forall L_u \in O_i$:
1. compute Π_C^* based on flags f^k;	1. $\forall$ paths $P_i^k \in \{P_i^k\} \vert P_i^k \rightarrow L_u \in Resv^u \setminus \Pi^*$:
2. update Π^* based on flags d^k;	set $d^k = 0$ if $b_C^k > \rho'_u$,
3. if $\Pi^* = \emptyset$ or $\Pi^* = \Pi_C^*$, return Π^*.	where the available output bandwidth ρ'_u
4. else update network view $\mathcal{N}'_S$	is updated according to a greedy allocation;
send $Path^u$, $\forall L_u \in O_S$.	2. send $Resv^v$, $\forall L_v \in I_i$.
node N_i:	client C:
upon receive $Path^u$, $\forall L_u \in I_i$:	upon receive $Path^u$, $\forall L_u \in I_C$:
1. update network graph $\mathcal{N}'_i$	1. compute the set of available paths Π_C;
2. compute available paths Π_i according to $\mathcal{N}'_i$;	2. compute the optimal allocation Π_C^* from Π_C;
3. compute extended paths Ω_i, resp. Γ_i, $\forall L_v \in O_i$;	3. $\forall P_C^k \in \Pi_C^*$, set $f^k = d^k = 1$;
4. send discovery messages $Path^v$, $\forall L_v \in O_i$.	4. send reservation messages $Resv^v$, $\forall L_v \in I_C$.

successfully reserved by S (i.e., all d^k flags are set to 1), the nodes update their local view of the network, $\mathcal{N}'_i = \mathcal{N}_i \setminus \Pi^*$, 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 optimal rate allocation, which is reached when all reserved flows by C can be accommodated by the network at the requested rate b_C^k.

The path extension rule directly controls the convergence to the stable rate allocation, but also the quality of the rate allocation. Comprehensive information about end-to-end paths as created by Rule 1 allows to reach an optimal rate allocation, but at the expense of possible several iterations of the path reservation schemes. The algorithm however converges in a small number of rounds to a feasible solution, 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 allows for a quicker computation of the solution, which may however be suboptimal. Both algorithms are analyzed in Section V and their performance is compared in Section VI.

The distributed path selection and rate allocation algorithms are summarized in Algorithm 1, where the left-hand side, and right-hand side columns respectively correspond to the path discovery, and path extensions phases. The algorithms differ in the path extension rule (step 3 in the bottom left block). Initially, both algorithms start at the server side, with Step 4. For the sake of clarity, we call Algorithm 1, resp. Algorithm 2, the distributed path allocation and rate allocation solutions that rely on Rule 1, resp. Rule 2 for path extension.

V. Analysis and Discussion

A. 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 one run of our algorithms, we derive hard bounds on the convergence of the rate allocation towards the optimized solution. Observe that one round of the algorithms requires one 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 assumption about the stability of the network in terms of average bandwidth and loss probability of the network links is therefore generally valid since the rate allocation algorithms converge in a very small number of steps, as shown in the next section. Since the total number of paths is quite small in general [21], the algorithms reach a stable solution after a convergence time that corresponds to only a few RTTs, during which the average link characteristics are likely to stay unchanged.

We consider first the Algorithm 1, which uses Rule 1 for path extension, so that the client has a complete view of end-to-end paths to compute the path selection. We show that the Algorithm 1 converges 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 to 3. If b_C^k represents the available rate of on requested path $P_C^k \in \Pi_C^*$, we have $b_C^k \leq \rho_u$, $\forall L_u \in P_C^k$. Since, by hypothesis, the chosen paths P_C^k do not contain any joint bottleneck link L_u, we have $\rho_u \geq \sum_{k:L_u \in P_C^k} b_C^k$, $\forall L_u \in P_C^k$ 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^i_C between S and C which is affected by the smallest loss probability p^i_C.

Proof: Let $P^i_C \in \Pi^*_C \setminus \Pi^*$ be the lowest loss probability path requested by C but not yet reserved by our algorithm. Observe that P^i_C 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^i_C, the flow P^i_C will have priority during the greedy reservation phase of Algorithm 1.

Indeed, from the path extension operation we have $b^i_C \leq \rho_u$, $\forall L_u \in P^i_C$. Hence, P^i_C is successfully reserved at each intermediate node N_i on the path. Finally, the flow P^i_C 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: $\Pi^* = \Pi^* \cup P^i_C$. ■

Property 3. Algorithm 1 converges, and terminates in at most m rounds, where m is the number of allocated flows, that is 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, on subsequent rounds of the algorithm, the client C will not be able to request an infinite number of flows. ■

The previous properties show that 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. And in general networks, 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 $\Pi^*_C = \{P^1_C \ldots, P^m_C\}$ be the optimal set of paths chosen by C for transmission according to Theorems 1 to 3, based on the information received from the network nodes. Let finally b^k_C be the rate of the requested path $P^k_C \in \Pi^*_C$, with $b^k_C \leq \rho_u$, $\forall L_u \in P^k_C$. 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 $\sum_{k:L_u \in P^k_C} b^k_C \leq \rho_u$. This means that any node N_i, upon the 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 all available end-to-end paths between S and C are disjoint. The rate allocation operations during path extension in the 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. ■

B. 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 [22]. Since full knowledge about network status cannot be achieved in large scale networks, distributed path computation solutions are certainly advisable. They additionally allow to release the computational burden of a single node/server, and distribute it among several intermediate nodes [18].

In this paper, 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 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 streaming process 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 round-trip times. Hence, running our algorithm periodically, on timescales equal to the network estimation intervals ensures the optimal transmission decision, with the latest estimation about the network state.

We now identify a few typical scenarios where optimal 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.

- 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 8 non-overlapping channels according to the IEEE 802.11a standard specifications), and by optimizing the transmission schedule in the wireless network [24]. For example, the authors of [25] test a protocol stack that allows one wireless network card to be simultaneously connected to, and switch between, multiple WLANs in a transparent way for the application. In the same time, the authors of [26] present a video system over WLANs that uses multiple antennas in order to aggregate the rate of multiple wireless channels. Our decentralized algorithms ensure the end-to-end path computation at each intermediate node, taking into account the requirements of the media application.
- 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. Existing commercial products [27] can already maintain connectivity to multiple wireless services, and transparently switch at any time to the service that offers the best channel performance, for a fixed subscription price. It is only a question of time before commercial products will be able to aggregate the resources of multiple services in order to enhance the user streaming experience, and telecommunications operators are actively working on such systems. In this scenario, our decentralized algorithms can compute the parameters of the various end-to-end paths to the client, over the various offered network services.
- Overlay Network Scenarios (e.g., 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 from multiple transmission flows employed by the application. Our proposed algorithms can be applied directly to find the end-to-end paths from the server to the client, via multiple intermediate overlay nodes. The discovered paths are later used by the client to compute the optimal transmission flows along with their rate allocation.

VI. Simulations

A. 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, realistic scenario, implemented in ns2 [28] 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 of 50 seconds 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 [29]. 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.

B. 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 rates, low loss probabilities), 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, with 10 nodes each. Any two nodes 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 the wired and wireless links are presented in Table I.

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 [21]. The results for both network scenarios are presented in Figure 2. We observe that the great 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 Figure 3 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

TABLE I

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}	$3 \cdot 10^6 bps$	$7 \cdot 10^5 bps$
p_{min}	10^{-4}	10^{-3}
p_{max}	$5 \cdot 10^{-3}$	$4 \cdot 10^{-2}$

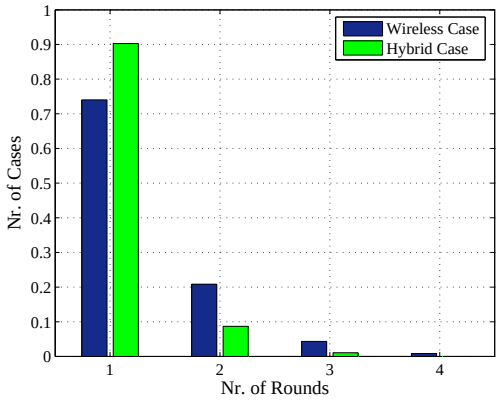

Fig. 2. Number of rounds of the iterative rate allocation, necessary to converge to optimal solution of Algorithm 1.

decreases, and that the partial 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 Figure 2 and Figure 3, 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 which 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. The cumulative density function is represented in Figure 4, which illustrates the probability for the improvement in quality to be within a predefined range $[0, x]$. We observe that, for both network scenarios, our algorithm obtains significantly better results in more that 70% 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 Figure 5, where we observe

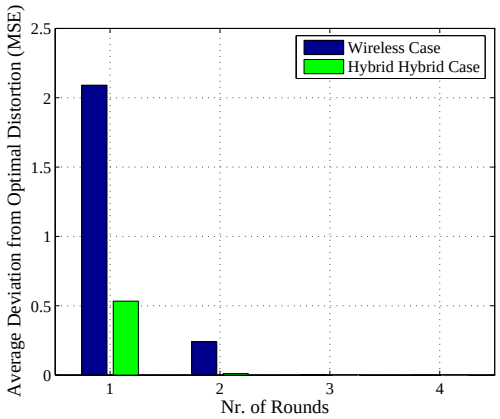

Fig. 3. Convergence of Algorithm 1, measured in terms of video distortion (MSE) as compared to the optimal solution.

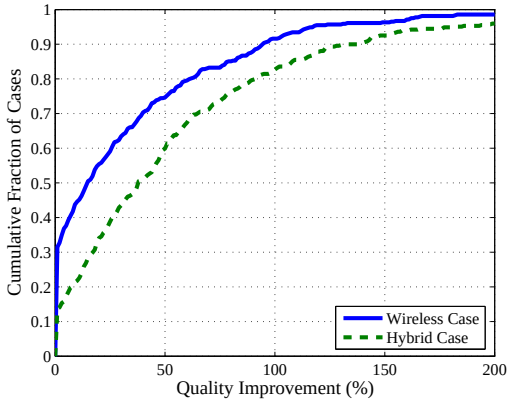

Fig. 4. Cumulative density function for the improvement in quality offered by Algorithm 1 vs. a Heuristic Rate Allocation Algorithm.

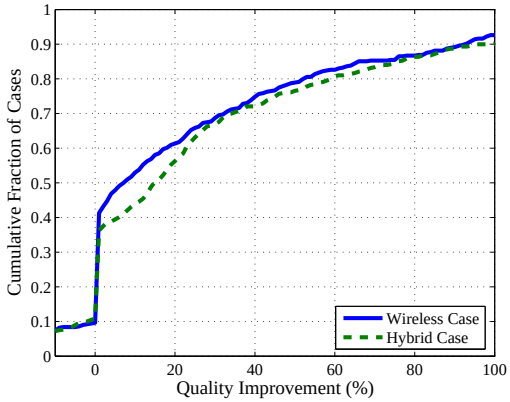

Fig. 5. Cumulative density function for the improvement in quality offered by Algorithm 2 vs. a Heuristic Rate Allocation Algorithm.

that Algorithm 2 also performs much better than a 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 Figure 6 and Figure 7. Figure 6 represents the cumulative density function of the difference incurred by Algorithm 2, with respect to the optimal allocation offered by Algorithm 1. A similar representation is proposed in Figure 7, except that 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 Figure 6. From both figures, we see that, for the $Wireless$ scenario, the performance of the greedy scheme is equal to the optimal solution in almost 65% of the cases. Algorithm 2 is even better, when compared to the execution of the optimal algorithm after the first round (70% of the cases providing equal or better 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. Good results are also observed for the $Hybrid$ case. However, in this case 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 flows chosen for the streaming application. The results for the $Wireless$ and $Hybrid$ network scenarios are presented in Figure 8 and Figure 9, 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 II 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

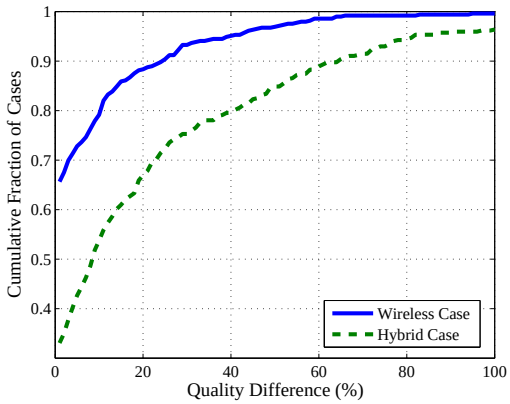

Fig. 6. Cumulative density function of the relative difference in quality, for Algorithm 1 vs Algorithm 2.

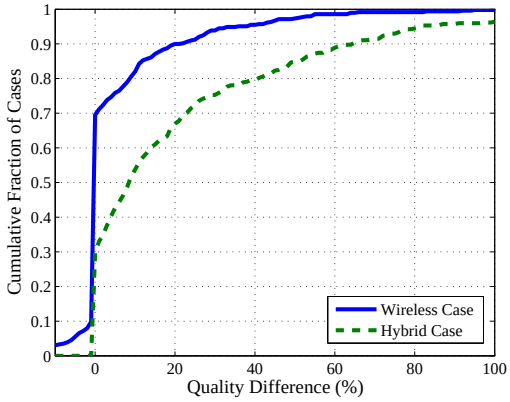

Fig. 7. Cumulative density function of the relative difference in quality, for Algorithm 1 limited to one iteration only, vs Algorithm 2.

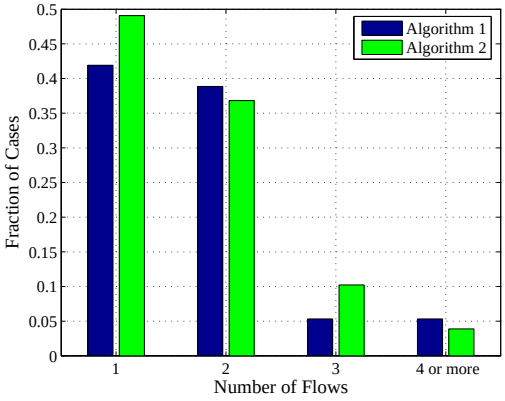

Fig. 8. Average Number of Flows used by Algorithms 1 and 2 in the $Wireless$ Network Case.

TABLE II

AVERAGE TRANSMISSION RATES CHOSEN BY ALGORITHMS 1 AND 2

	$Wireless$	$Hybrid$
Algorithm 1	$531kbps$	$797kpbs$
Algorithm 2	$473kbps$	$591kpbs$

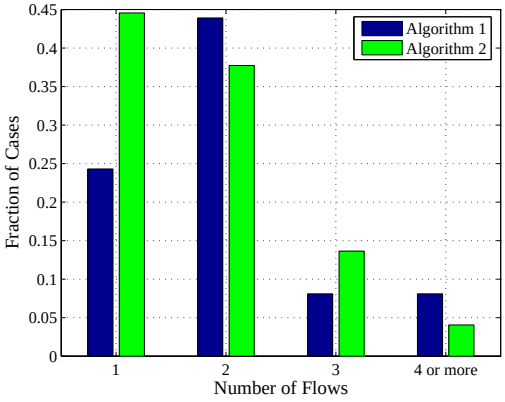

Fig. 9. Average Number of Flows used by Algorithms 1 and 2 in the $Hybrid$ Network Case.

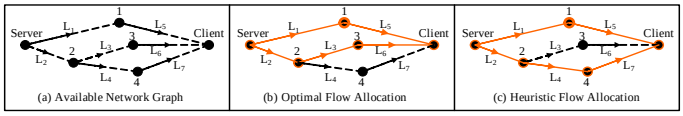

Fig. 10. Network scenario: a) Available network graph; b) Flow allocation chosen by Algorithm 1; c) Flow allocation chosen by Algorithm 2.

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

C. Specific 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. We send the $foreman$ sequence, encoded at $375kbps$ and $550kbps$ over a network as presented in Figure 10 (a). The network scenario is reproduced in the ns2 simulator, and the path computation mechanisms 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 models with exponential distribution of staying time, and average rates between 100 and $300kbps$. 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 III, for the two cases under consideration. The network MTU is set to 1000 bytes worth of video data. Finally, we consider cases where the video stream is sent with, respectively without forward error protection. Forward error protection employs FEC codes with blocks (20,18) and (20,19) for the first and second network case, respectively. We assume that all video packets can be recovered if at least 18, respectively 19 packets are correctly received in a block of 20 packets.

Figure 10 b) and c) first show the path selection provided by Algorithm 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 allocation, and each simulation point is averaged over 10 simulation runs. Figure 11 and Figure 12 present the performance of Algorithms 1 and 2 as a function of the playback delay imposed by the client, respectively in presence of absence 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 allows 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 losses, and late arrivals at the client. We observe that, even if the choice of transmission paths differs between the two algorithms,

TABLE III
PARAMETER VALUES FOR THE NETWORK LINKS IN FIGURE 10

	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

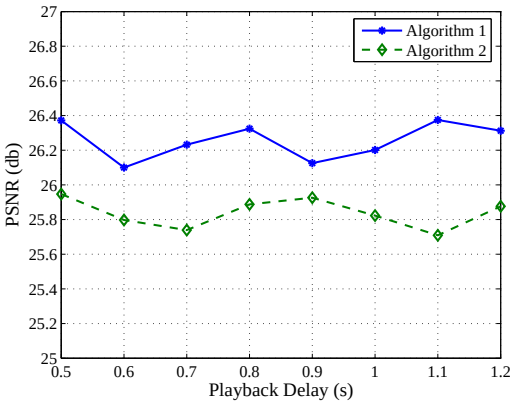

Fig. 11. Performance evaluation of Algorithms 1 and 2 as a function of playback delay (Network Case 1, no FEC).

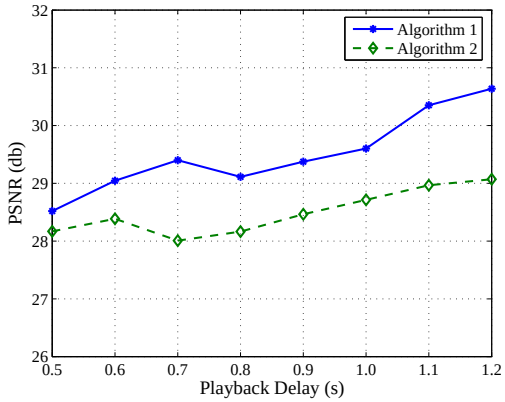

Fig. 12. Performance evaluation of Algorithms 1 and 2 as a function of playback delay (Network Case 1, with FEC).

the performance is similar, since the end-to-end paths are disjoint, and quite homogeneous in the network case under study. It can be noted that the influence of the playback delay is similar for both schemes. In the same time, it can be observed that using even a minimum error protection strategy unsurprisingly improves the final results. While simple distortion models that encompass the effect of channel protection can be proposed as an extension to the rate allocation strategy, the design of optimal joint source and channel coding strategies are however beyond the scope of the present paper. 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 run for each algorithm, and analyze the temporal evolution of the quality. The reconstructed video quality is measured at the receiver for each group of 30 pictures, in the absence and presence of FEC, respectively. Results are presented in Figure 13 and Figure 14 for the second network case, where the playback delay imposed by the client is set to one second. It can be seen that both algorithms again perform similarly in the presence of network losses and cross traffic. It confirms the results presented above, and positions both algorithms as interesting solutions for media-specific rate allocation in multipath networks.

VII. 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 available set of transmission paths to the client is created only from the original local network views at each individual intermediate node. It allows to release the assumption of full network knowledge at any single node in the network

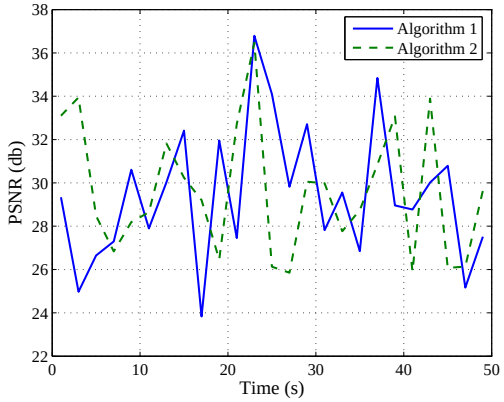

Fig. 13. Temporal evolution of the video quality (Network Case 2, no FEC).

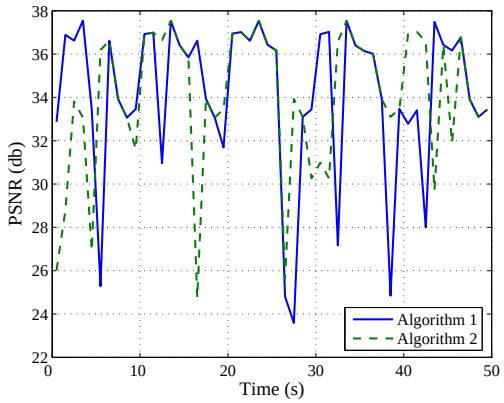

Fig. 14. Temporal evolution of the video quality (Network Case 2, with FEC).

and eliminates the need for expensive path monitoring mechanisms. Both solutions therefore represent interesting alternatives for media specific path selection in large scale networks. In particular, extensive simulations demonstrate that the optimal algorithm converges very fast, in particular in networks that present a small number of bottleneck links. In 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] L. Golubchik, J. Lui, T. Tung, A. Chow, and W. Lee, "Multi-path continuous media streaming: What are the benefits?" *ACM Journal of Performance Evaluation*, vol. 49, no. 1-4, pp. 429–449, Sept 2002.

[2] Y. Li, S. Mao, and S. S. Panwar, "The case for multipath multimedia transport over wireless ad hoc networks," in *Proceedings of IEEE/ACM BroadNets*, October 2004, pp. 486–495.

[3] D. Jurca and P. Frossard, "Media-specific rate allocation in multipath networks," *IEEE Transactions on Multimedia*, 2006, submitted.

[4] J. L. Sobrinho, "Algebra and algorithms for qos path computation and hop-by-hop routing in the internet," *Journal of IEEE/ACM Transactions on Netowrking (TON)*, vol. 10, no. 4, pp. 541–550, August 2002.

[5] T. Murakami, M. Bandai, and I. Sasase, "Split multi-path routing protocol with load balancing policy (smr-lb) to improve tcp performance in mobile ad-hoc networks," *Journal of IEICE Transactions on Communications*, vol. E89-B, no. 5, pp. 1517–1525, 2006.

[6] S. Vutukury and J. J. Garcia-Luna-Aceves, "Mdva: A distance-vector multipath routing protocol," in *Proceedings of IEEE INFOCOM*, vol. 1, 2001, pp. 557–564.

[7] Y. Cui, Y. Xue, and K. Nahrstedt, "Optimal resource allocation in overlay multicast," *Journal of IEEE Transactions on Parallel and Distributed Systems*, vol. 17, no. 8, pp. 808–823, August 2006.

[8] S. Sarkar and L. Tassiulas, "A framework for routing and congestion control for multicast information flows," *IEEE Transactions on Information Theory*, vol. 48, no. 10, pp. 2690–2708, October 2002.

[9] W. Wei and A. Zakhor, "Robust multipath source routing protocol (rmpsr) for video communication over wireless ad-hoc networks," in *Proceedings of IEEE International Conference on Multimedia and Expo' (ICME2004)*, 2004.

[10] S. Savage, A. Collins, and E. Hoffman, "The end-to-end effects of internet path selection," in *Proceedings of ACM SIGCOMM*, 1999, pp. 289–299.

[11] T. Nguyen and 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.

[12] S. Mao, S. Lin, S. S. Panwar, Y. Wang, and E. Celebi, "Video transport over ad hoc networks: Multistream coding with multipath transport," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 10, pp. 1721–1737, December 2003.

[13] A. C. Begen, Y. Altunbasak, O. Ergun, and M. H. Ammar, "Multi-path selection for multiple description video streaming over overlay networks," *Signal Processing: Image Communcation*, vol. 20, pp. 39–60, 2005.

[14] J. Kim, R. M. Mersereau, and Y. Altunbasak, "Network-adaptive video streaming using multiple description coding and path diversity," in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, 2003.
[15] X. Zhu and B. Girod, "Distributed rate allocation for multi-stream video transmission over ad-hoc networks," in *Proceedings of IEEE ICIP*, 2005.
[16] T. Nguyen and A. Zakhor, "Multiple sender distributed video streaming," *IEEE Transactions on Multimedia*, vol. 6, no. 2, pp. 315–326, April 2004.
[17] S. Tao and R. Guerin, "Application-specific path switching: A case study for streaming video," in *Proceedings of ACM Multimedia*, October 2004, pp. 136–143.
[18] N. Lynch, *Distributed Algorithms*. San Mateo, CA: Morgan Kaufmann Publishers Inc., 1996.
[19] R. Rodrigues, B. Liskov, and L. Shrira, "The design of a robust peer-to-peer system," in *Proceedings of SIGOPS European Workshop*, 2002.
[20] D. L. Mills, "Network time protocol implementation, specification and analysis," Network Working Report RFC 1305, Tech. Rep., March 1992.
[21] D. Jurca and P. Frossard, "Media-specific rate allocation in heterogeneous wireless networks," in *Proceedings of the International Packet Video Workshop*, 2006.
[22] http://www.icir.org/models/tools.html.
[23] V. Ribeiro, R. Riedi, R. Baraniuk, J. Navratil, and L. Cottrell, "pathChirp: Efficient available bandwidth estimation for network paths," in *Proceedings of Passive and Active Measurement Workshop* , April 2003.
[24] P. von Rickenbach, S. Schmid, R. Wattenhofer, and A. Zollinger, "A robust interference model for wireless ad-hoc networks," in *Proceedings of IEEE WMAN'05*, 2005.
[25] R. Chandra, P. Bahl, and P. Bahl, "Multinet: Connecting to multiple ieee 802.11 networks using a single wireless card," in *Proceedings of IEEE INFOCOM*, vol. 2, 2005, pp. 882–893.
[26] G. Epshtein, "A better way to implement video over wlan," in *Metalink - white paper*, 2005.
[27] Swisscom Mobile Unlimited UMTS/GPRS/WLAN, http://www.swisscom-mobile.ch/scm/gek_mobile-unlimited-en.aspx.
[28] http://www.isi.edu/nsnam/ns/.
[29] D. Jurca and P. Frossard, "Video packet selection and scheduling for multipath streaming," *IEEE Transactions on Multimedia*, 2006, accepted for publication.