INTERNATIONAL JOURNAL OF COMMUNICATION SYSTEMS
Int. J. Commun. Syst. (2015)
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/dac.2919

Using topology information for quality-aware Peer-to-Peer multilayer video streaming

Niels Sluijs, Tim Wauters, Chris Develder, Filip De Turck, Piet Demeester and Bart Dhoedt*,†

Department of Information Technology (INTEC), Ghent University – iMinds, Gaston Crommenlaan 8, Bus 201, 9050, Ghent, Belgium

SUMMARY

Due to increasing bandwidth capacities, the Internet has become a viable transport medium for a (live) video. Often, delivery of video streams relies on the client–server paradigm and therefore exhibits limited scalability. The Peer-to-Peer (P2P) network model is an attractive and scalable candidate to stream video content to end users. However, these P2P frameworks typically operate in a network agnostic mode. Introducing network topology information into these P2P frameworks offers opportunities to enhance the performance.
In this paper, we introduce a model to include network information when streaming a (multilayered) video in P2P frameworks. An important metric for video stream providers is the content quality perceived by end users. The optimization studied here aims at maximizing the number of users receiving a high quality video.
The paper addresses the optimization problem seen from the stream provider's viewpoint, having access to network topology information. An exact optimization approach is presented for benchmarking purposes and a heuristic approach to cope with realistic network sizes. In addition, we present an approach to decide the deployment location of peering functionality. The results show that our strategy significantly decreases the fraction of destinations receiving only the base layer, and by introducing extra peering functionality, network capacities are used more efficiently. Copyright © 2015 John Wiley & Sons, Ltd.

Received 5 November 2013; Revised 25 November 2014; Accepted 25 November 2014

KEY WORDS: dimensioning; heuristic optimization; integer linear programming; multilayer video streaming; Peer-to-Peer

1. INTRODUCTION

Delivery of live video streams is often realized through client–server systems: servers send video feeds directly via dedicated networks to end users. Due to growing bandwidth capacities (especially in the access networks), the Internet gets a more prominent role as being the main medium for transporting (live) video streams to millions of heterogeneous end-devices. Websites of, for example, YouTube and EUROVISION Sports are already exploring the possibilities of broadcasting popular (live) events all over the world such as World Championships and the Olympic Games. Additionally, different television broadcasting companies already offer streaming services to watch (live) television programs via their websites (e.g., British Broadcasting Corporation iPlayer (BBC Broadcasting House, Portland Place, London, UK), and RTL XL (RTL Group, Luxembourg, Luxembourg)). However, often, the quality of the stream and the total number of viewers at the same time are still limited because bandwidth capacities on the broadcasting servers form the main bottleneck. An interesting network model that offers a scalable mechanism for distributing a (live)

*Correspondence to: Bart Dhoedt, Department of Information Technology (INTEC), Ghent University – IBBT, Gaston Crommenlaan 8, Bus 201, 9050, Ghent, Belgium.
†E-mail: Bart.Dhoedt@intec.ugent.be

video is the Peer-to-Peer (P2P) overlay technology. In a P2P network, the peers form a virtual overlay network on top of an actual network and all peers act as both supplier and consumers (contrary to traditional client–server networks). Because peers can supply downloaded data to each other (i.e., parts of the video stream), not only a scalable and robust solution is offered but also the server loads on the original source nodes (i.e., injector node) are reduced.

In order to optimally adapt to typical heterogeneous circumstances (such as various asymmetric link bandwidths, end-devices having different display resolutions, or even stereoscopic rendering possibilities), next generation P2P video streaming networks use a multilayered video [1, 2] (e.g., scalable video coding [3]). The video is encoded into multiple layers (i.e., usually divided into a temporal, spatial or quality resolution, or a combination of the three) and allows playback of the video when only the base layer is received. Every additional received video layer increases the user's experienced viewing quality of the video feed (such as an increased frame rate or the resolution scaled from e.g., 720 p to 1080 p). Using multilayered video coding, end-devices can choose to download only the video layers that they are able to output (e.g., based on screen resolution or stereoscopic rendering abilities). Even when two nodes are choosing to receive a different video quality, using a multilayer video has the advantage that both nodes have the ability to exchange video layers. The bandwidth requirements for the node inserting the (live) video stream into the network can be mitigated significantly, because a single stream (containing each distinct video layer) might already be sufficient to allow each device to select the right number of video layers to stream.

In this paper, we study the live video stream distribution problem from a service provider's perspective. The service provider has to optimize the stream delivery, such that the stream quality received by the end users is optimal. To this end, the service provider installs peering nodes on well-chosen locations in the network and leases sufficient network capacity to interconnect these peering nodes. Consequently, the service provider has a good view on the resulting peering node and network infrastructure and is faced with the problem to properly configure this infrastructure for a high-quality stream delivery.

The proposed-P2P framework to transport multilayer video, as illustrated in Figure 1, consists of the following entities:

- *Injector node*: offers the video stream to the rest of the network, separated into distinct video layers.
- *Tracker node*: a (often centralized) bootstrap server, providing all necessary information for peering nodes to start the download process. When the streaming of a video is started, the tracker node has a coordinating role.
- *Peering node*: nodes containing the peering software functionality and (usually) acting as the entry points for end-devices (i.e., *destination nodes*) to get the video stream from. In this paper, we use the term peering node and peer interchangeably.
- *Forwarding node*: nodes that forward data through the network and are usually located in the core network (i.e., underlay network topology).

An important aspect when using multilayer video coding in a P2P network is the piece picking and peer selection mechanism. Current strategies (e.g., Tribler [4]) try to maximize the local peer's video (i.e., download) quality in a greedy fashion, without considering other peer's quality. However, for a video streaming network to become successful, an important aspect is to optimize the overall received video quality in the network. When peering nodes would collaborate, the average delivered video quality can be increased at the expense of reducing the quality of a few peers receiving the stream at higher than average quality. When these peers decide to give up a bit of their high video quality (i.e., drop a few of the top video layers), more bandwidth and (possible) lower layer quality pieces will become available to the rest of the network. Peers streaming at a possibly much lower video quality benefits from this strategy and are provided a chance to increase their streaming quality to acceptable levels. The strategy we propose in this paper maximizes the minimum received video quality at each destination by using topology information of the actual network to orchestrate peer selection and thereby forming a future proof and robust framework for distributing (live) video streams over the Internet.

DOI: 10.1002/dac

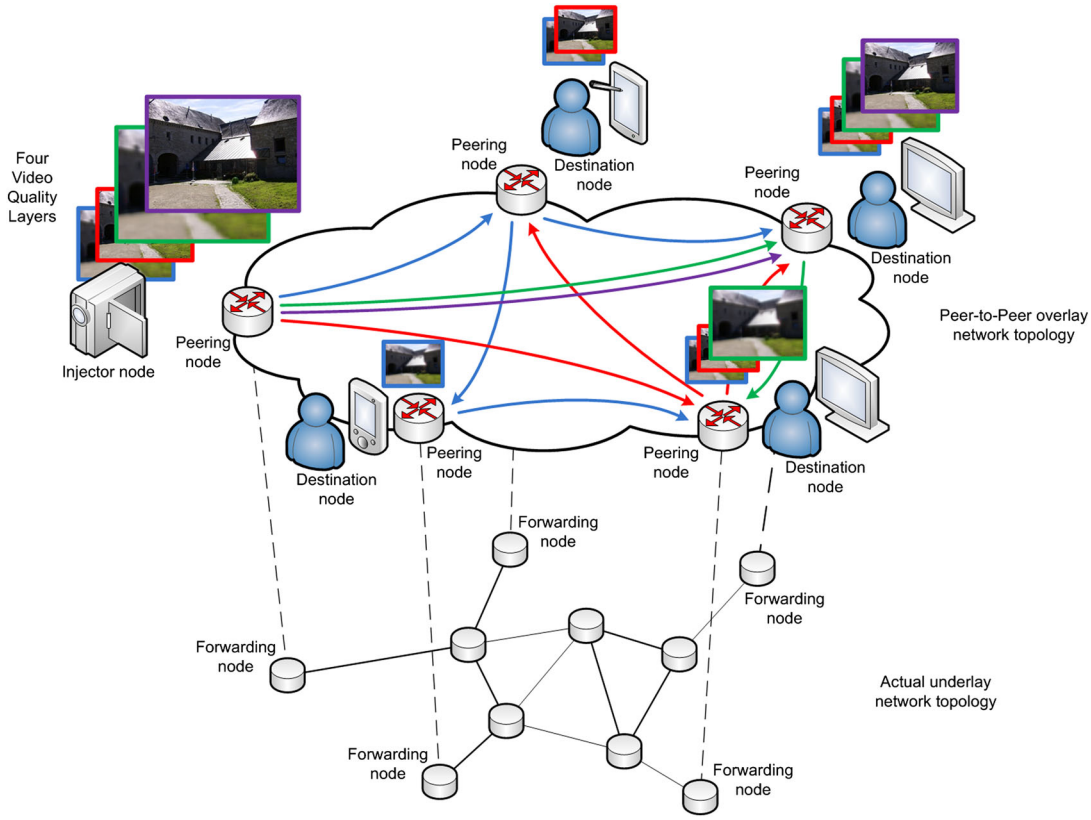

Figure 1. Next generation Peer-to-Peer (live) video streaming network uses multilayer video coding, which allows to start watching a video when only the base layer is downloaded. Additional received layers increase the video quality, and this strategy allows peers to adopt to their output abilities and the network to offer a higher average (or minimal) received video quality to end users.

As a used case, we study the possibility of using a P2P multilayer video framework to transport a (live) video feed from a video service provider's (VSP) perspective, which has a precise view on the underlying network topology. Additionally, we study the situation where VSPs in combination with Internet service providers are willing to make relatively small hardware investments in order to turn their networks into a good alternative for distributing (live) video streams (i.e., by upgrading nodes in their network to exhibit peering application intelligence). Because VSPs should be able to guarantee a certain minimum quality level, optimizing this minimum level is of key importance in this respect.

To optimize the received video quality, we construct an Integer Linear Programming (ILP) problem formulation that describes the underlay-overlay-routing problem of the video layers in the network. By using an exact solver, we are able to study different objective strategies (i.e., the traditional selfish and our proposed social download strategy) for relatively small network topologies. To study relatively larger network sizes (or more complex problems), we propose a heuristic strategy to compute both the underlay and (consequently) overlay routing for a live video stream. Using the results obtained from the exact solver, we can validate our heuristic method. Additionally, we extend our heuristic optimization method to calculate ideal positions to place peering node functionality, which allows VSP/Internet service provider to dimension their network topologies properly.

The proposed methods can be used by an (central) orchestration engine that controls the (overlay) connections between peers, managing the number of video layers received by a destination in function to increase the number of video layers for multiple other destinations. Although this orchestrating unit would seem a single-point-of-failure in the proposed architecture, we assume that this management unit is realized on top of existing solutions (e.g., [4, 5]) such that default routing would be enabled in case the orchestration engine would fail. An alternative approach would be to

DOI: 10.1002/dac

implement the orchestration engine in a distributed fashion, where deployment of different versions of the same component ensures robustness against failures.

The core network topology used in this paper is illustrated in Figure 2, which is loosely based on the GÉANT backbone topology (GÉANT, Cambridge, UK) in June 2011 [6]. Although complex models exist to generate (background) traffic representing different kinds of applications (e.g., general P2P software or content distribution networks) [7], we assume that the service provider has leased network capacities; expressed here as a number of video layers that can be transported on each link, assuming the same constant bandwidth usage for each video layer. In our used case, the (live) video stream is inserted from the European Parliament located in Strasbourg, France (abbreviated as FR). End users requesting the (live) video feed are modeled by connecting a $<$ peering, destination $>$ node couple to a country's forwarding node. To study the effects of asymmetrical bandwidth capacities, which is one of the currently limiting factors for the overall performance of P2P networks, the peering node's uplink (i.e., the link from the peering node to the country's forwarding node) is limited to half of the average capacity among all links incident to the corresponding forwarding node. For instance, the uplink of the peering node at Iceland (abbreviated as IS in Figure 2) has a maximum bandwidth capacity of three video layers. We assume that at most one $<$ peering, destination $>$ node pair is connected to a country's forwarding node.

The paper continues in section 2 with an overview of related work. Section 3 provides the generic problem formulation and validates the model theoretically for a tree-like network topology. Section 4 solves the problem as explained previously, applying an exact solver. Section 5 provides and validates our heuristic method for solving the routing of video layers in a network. An extension of our heuristic optimization strategy is given in section 6 that calculates the locations in the network to place peering node functionality. Conclusions and future work are presented in section 7.

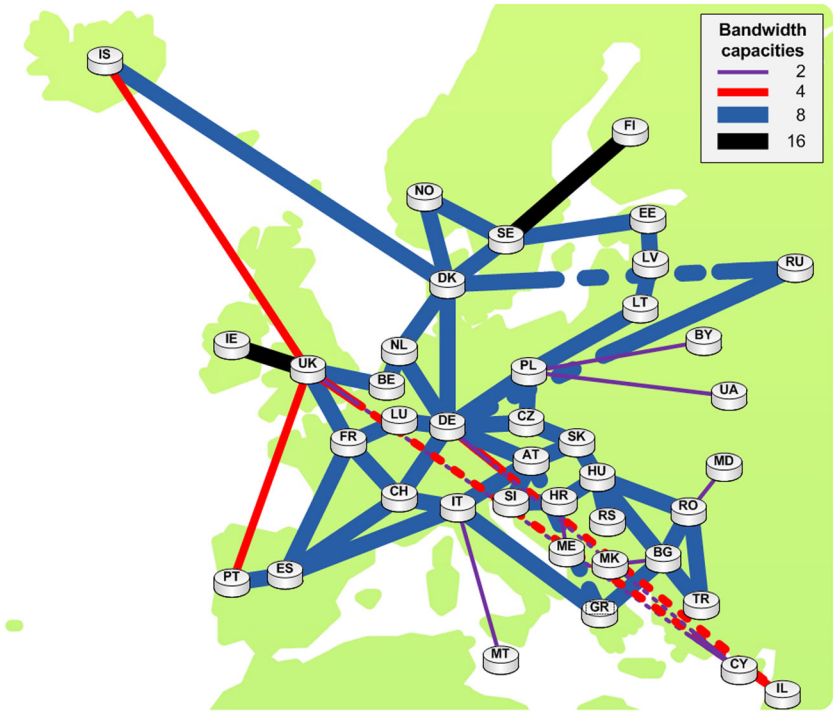

Figure 2. Mesh-based core network topology, inspired by the GÉANT backbone topology. We consider the injector node, inserting the multilayer (live) video feed from the European Parliament to be located in France. The bandwidth capacities represent the maximum number of video layers the link is allowed to carry, in each direction separately. We assume that each video layer has the same constant bandwidth cost, and dashed lines are used to indicate non-intersecting links.

DOI: 10.1002/dac

2. RELATED WORK

A number of commercial platforms exist that offer streaming video using the Internet as a transport medium (e.g., Octoshape, RawFlow, and RayV). However, often, traditional client–server-based network models are used to transport the video data (e.g., [8]), resulting in a large server and bandwidth costs for broadcasters. On the other hand, several freeware/open-source frameworks employ P2P techniques to offer (live) video streaming solutions, for example, Alluvium, End System Multicast, PeerCast, PPTV (formerly known as PPLive), and Tribler [4]. In order to get stable streams with acceptable quality, large numbers of end users are required who watch the same video feed. To the best of our knowledge, none of the P2P (multilayer) video streaming networks use topology information to optimize the load in the network.

Another option would be to use Internet Protocol (IP) multicast [9, 10]. However, IP multicast is not a feasible solution for a scalable video streaming service because of the sparse deployment on the Internet. Therefore, our solutions focus on advanced P2P techniques, without requiring substantial hardware changes to deploy the designed frameworks.

Inherent to P2P networks are mechanisms that motivate/force peers to share with each other when downloading such as BITTORRENT's tit-for-tat mechanism (BitTorrent Inc., San Francisco, CA, USA) [11] and Tribler's give-to-get algorithm [12]. An unbalanced data exchange is a problem that decreases a P2P system's potential performance in terms of bandwidth utilization [13–16]. Both the unwillingness to share by users and inferior data exchanges as a result of the distributed algorithms form a huge challenge when designing a P2P streaming applications. Therefore, incentives mechanisms are necessary to allow successful deployment of robust and a scalable (live) video streaming framework. Our approach to solve these issues is based on extending tracker node privileges with an (distributed) orchestrating function, managing all data transfers in the network. The aim of this engine is to compute an optimal routing strategy for the full network and to provide the highest possible performance gain.

Studies combining P2P technologies with layered video are performed in References 17–19 and mainly focus on advanced and selfish buffering strategies by altering the piece picking algorithms and/or neighbor selection strategies. Our solution complements that work by using topology information to manage the video streaming process and by aiming to guarantee a minimum quality level for all stream consumers.

An advanced caching strategy is proposed by Lin and Lee 20 to effectively utilize video buffer space brought into the network by proxy servers (forming content distribution networks). Each peer is streaming the video (via a proxy) and buffers a (limited) set of the video segments. The strategy involves chaining peers that is streaming the same video blocks thereby allowing the proxy server to cache other video blocks that cannot be fetched by any of the collaborating peers. Although the caching scheme of Lin and Lee 20 is not network topology awareness, the strategy encompasses caching multilayered video to further alleviate (proxy) server requirements and therefore is an interesting extension when calculating ideal positions to place peering node functionality.

In Reference 21, a P2P video-on-demand strategy to optimally (pre) fetch video segments is presented, integrating localization and congestion-aware peer selection schemes. Simulation results show that utilizing location information (and preventing congestion) increases the average supported playback rate of a video. However, in Reference 21, multilayer video coding is not considered, and peers are classified to a fixed set of domains. Our current work also addresses multilayer video coding schemes thereby offering a whole range of additional optimization opportunities. Moreover, in contrast to Fouda *et al*. 21, we do not restrict a peer to be part of a fixed set of domains, which allows us to calculate the optimal solution.

In References 22–24, mathematical formulations are introduced to model video streaming using P2P mechanisms. In these studies, only overlay routing is considered, and the authors assume the upload and download capacities for each peer to be the main bottlenecks. In addition to overlay routing, our proposed optimization strategy takes the underlay network into account, allowing to take constraints into account imposed by shared network links in data exchanges. As a consequence, the complexity of the problem significantly increases but yields better solutions in terms of bandwidth usage and/or video stream quality.

DOI: 10.1002/dac

Capone *et al.* [25] study underlay and overlay routing optimization in combination with overlay node positioning to create virtual topologies on Internet-like networks, so-called service overlay networks (SON). Because the Internet was designed to provide best effort delivery, SON are used to provide end-to-end quality of service without requiring any modification to the underlying network infrastructure. Compared to Capone *et al*. 25, our proposed model considers the routing of multiple video layers to each destination, possibly via distinct overlay routes (including asymmetric bandwidth properties on the access links). Additionally, we require the video layers to be delivered in order and originated from the source (i.e., injector) node, resulting in a significant increase in the complexity to solve our problem.

Rigorously, solving the resulting multilayer video routing problem scales badly with the dimensions of the problem at hand. Therefore, we propose a heuristic approach, allowing to obtain solutions for larger problem instances, at the expense of loss of optimality. In this respect, several generic heuristic strategies exist that are used to find (almost) optimal solutions for various kinds of (combinatorial) problems such as ant colony optimization, Tabu search, genetic algorithms, and simulated annealing (SA). As a common denominator, all previously mentioned techniques use probabilistic approaches to prevent getting trapped in local optima. The problem we solve in this paper is in essence similar to the classic and heavily studied vehicle routing problem (VRP), where SA has proven to be a good candidate to solve computationally more complex versions of the standard VRP [26, 27]. Therefore, SA forms the basis for our global optimization strategy.

Contributions of this paper can be summarized as follows:

- Providing a mathematical formulation that is capable to model the underlay-overlay-routing problem of a multilayer video in a P2P network.
- Proposing a piece picking and peer selection strategies for next generation P2P video streaming networks that maximizes the minimum video quality for each destination, which is accomplished by orchestrating the download using a tracker node that has a precise view on both the underlay and overlay network topologies.
- Providing a stochastic heuristic optimization method to calculate the routing of a multilayered video in a P2P overlay network, taking into account the underlay topology.
- Using our heuristic strategy to find ideal positions to upgrade forwarding nodes to contain peering application functionality.

3. PROBLEM FORMULATION

Given a network topology, existing of one injector and multiple forwarding, destination and peering nodes, and bandwidth properties of the interconnecting links, our aim is to find the routing solution of all video layers in the underlay and overlay networks such that the chosen objective strategy is optimized. In order to find an optimal solution and to present a precise view on the problem, an ILP (Integer Linear Programming) formulation is given here. First, section 3.1 describes the network model, introducing all parameters and variables. Then, section 3.2 provides the formulation in terms of the objective function and a set of constraints that specify the relation between the parameters and the variables. Finally, section 3.3 shows the correctness of our model by comparing the results of solving our formulation on a basic network structure, with an analytical solution.

3.1. Model description

The problem can be characterized by the network topology, an injector node, forwarding nodes, peer nodes, destination nodes, and a list of video layers. Table I provides an overview of all symbols declared in sections 3.1.1.

3.1.1. The network. The underlay network is represented by a directed graph G, characterized by a set of nodes V of size $|V|$ and a set of directed edges E of size $|E|$. The graph is presumed to be bidirectional, but the edge properties can be asymmetric. Each directed edge e from E is characterized by a constant maximum bandwidth capacity $u_e \geq 0$. I_v and O_v are the sets of, respectively, ingoing and outgoing links of a node v. To enforce shortest-path routing, Dijkstra's algorithm is used to

DOI: 10.1002/dac

Table I. Symbols that define the problem solved by our optimization strategy

Symbol	Description
E	Set of all links in the network topology
u_e	Total bandwidth capacity of link $e \in E$
V	Set of all nodes in the network topology
I_v	All incoming links on node $v \in V$
O_v	All outgoing links from node $v \in V$
z	The node that inserts the (live) video stream into the network, $z \in V$
F	Set of forwarding nodes, $F \subset V$
P	Set of nodes containing peering application intelligence, $P \subset V$
D	Set of nodes where an end user is connected to, $D \subset V$
$m_{e,d}$	Binary constant parameter stating whether link $e \in E$ can be used to transport data to node $d \in D$
L	Set of distinct video layers
l_i	Video layer $l_i \in L$
b_i	Benefit value of receiving layer l_i
c_l	Bandwidth cost of transporting layer l
$h_{e,d,d',l}$	Binary variable indicating whether link e is used to carry traffic destined for node d, which is (in) directly sent to node d' and has layer l

compute the k-shortest-paths (measured in network hops) between node x and y. The constant value k controls the number of allowed paths between node x and y, for transporting video layers. Each link e is provided with a constant binary parameter $m_{e,d}$ that denotes whether or not link e can be used to transport data to node d using one of the k-shortest-paths. The value $m_{e,d}$ is 1 if and only if link e is on one of the k-shortest-paths to destination node d from a peer or injector node.

3.1.2. Forwarding and peer nodes. All network nodes that contain no application intelligence act as forwarding nodes. F is the set of all forwarding nodes, $F \subset V$ and has size $|F|$.

Peer nodes are the nodes running the P2P streaming application (i.e., an incoming stream can be sent to multiple destination peers) and are typically located at a user's home, connected via an asymmetric bandwidth connection to the Internet (i.e., rest of the network). P is the set of all peer nodes, $P \subset V$, $P \cap F = \emptyset$, and has size $|P|$.

3.1.3. Injector and destination nodes. The original source of the live video stream is provided by the injector node z. Destination nodes request the video stream and are usually connected to a peer node. D is the set of all destination nodes, $D \subset V$ and has a size of $|D|$.

3.1.4. Video layers. There is an ordered list of video layers L of size $|L|$, containing each different video layer sorted by increasing layer rank (i.e., layer l_0 is the base layer). Each layer l from L has a constant bandwidth cost per time unit of $c_l \geq 0$. Note that receiving a layer l_{i+1} is useless, without also receiving l_i, where $i \geq 0$. To keep the model simple, we consider only one scalability axis by representing the scalable video stream as one single (totally) ordered set of elements. However, the model can be adjusted to relieve this requirement or embed several scalability axes; the objective function (s) and constraint (14) have to be altered.

In order to prioritize for fairness situations where multiple destinations all receive a layer l_i over situations that only a few receive layer $l_{>i}$ and the rest receive $l_{<i}$, a constant value b_i per layer l_i is set to express the benefit of receiving the layer. The values $b_i = |D|^{(|L| - i - 1)}$ guarantee the following principle: $b_i > (D-1) \times \sum_{j=i+1}^{|L|-1} b_j$ for $0 \leq i < |L|$. This means that the benefit contribution b_i when a node receives layer l_i is greater than combining the benefits b_j for every other destination for all video layers l_j, where $j > i$.

3.1.5. Variables. In this subsection, e is an edge from set E, d and d' are destination nodes from set D, and l is a video layer from set L. The binary variable is as follows:

- $h_{e,d,d',l}$ is 1 iff edge e is used to carry traffic destined for node d, which is (in) directly sent to node d' and has layer l (with $|E| \cdot |D| \cdot |D| \cdot |L|$ h variables).

DOI: 10.1002/dac

Index d represents the destination node of the direct video traffic (i.e., underlay routing), and node d' indicates that another node can benefit from this traffic via P2P routing (i.e., indirect or overlay routing). The h variable is used to ensure flow conservation through the network and, where $d = d'$, allows the calculation of the bandwidth carried by the edges.

To illustrate the behavior of variable h, Figure 3 shows a scenario of a tree network with injector node z at its root. The video stream in this example consists of one layer and is transported over links that all have a bandwidth capacity of one layer. The matrices next to each edge store are the corresponding values for d and d' for the video layer. Two destination nodes (i.e., $D0$ and $D1$) receive the video layer and in between there are three peer nodes (i.e., P_z, P_{D0}, and P_{D1}) connected by a forwarding node (i.e., F). To ensure a correct solution, the injector node z sends the layer directly to node $D0$ and indirectly to node $D1$ over link a (i.e., another correct solution is sending the stream directly to node $D1$ and indirectly to node $D0$). Node P_z and F forward this traffic over, respectively, link c and e. Peer node P_{D0} duplicates the video layer and sends this traffic to destination node $D0$ and $D1$ via, respectively, links g and f. Forwarding node F and peer node P_{D1} forwards the direct traffic to $D1$ via links, respectively, links i and k.

This approach allows to perform both underlay and overlay routing through the network and to guarantee that the injector node z is the origin of every video layer that a destination node receives. However, the downside of this strategy is the increased complexity of solving the underlay-overlay-routing problem of the video layers and as a consequence limits the size of our network topologies.

3.2. *Formulation*

Now, that all symbols and variables are presented, the objective function to optimize can be formulated as follows:

$$Q = \text{maximum}\left(\sum_{l_i \in L}\sum_{d \in D}\sum_{e \in I_d} b_i \times h_{e,d,d,l_i}\right) \tag{1}$$

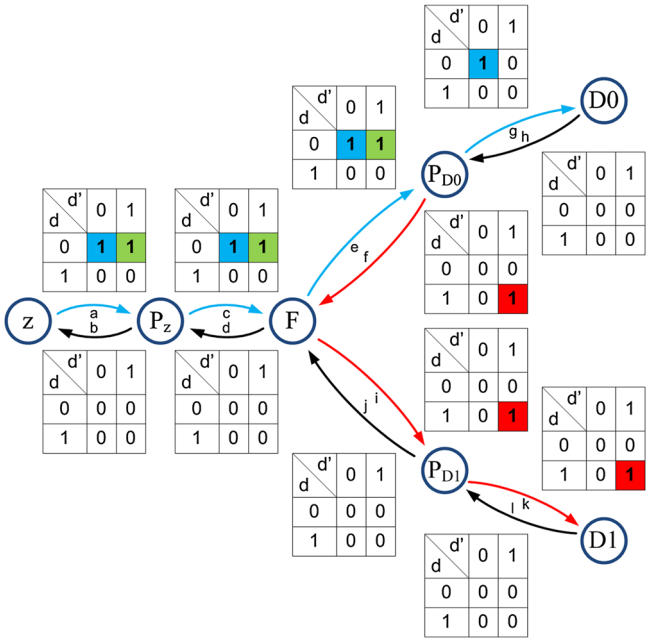

Figure 3. Usage scenario of the $h_{e,d,d',l}$ variables in the ILP formulation. Node z inserts a video consisting of one quality layer into the network. Because all the network links have a capacity to transport one layer, the direct traffic is sent from the injector node z to destination node $D0$. The peering node directly connected to $D0$ duplicates the video stream and also sends it to destination $D1$.

DOI: 10.1002/dac

By optimizing objective function Q, each node receives a maximum video quality, without requiring any other node in the network to lower its receiving quality. Hereby, the minimum quality that is received by each node is maximized. In order to solve the problem, a set of constraints has to be considered to make sure the relation between all parameters and variables comply with the general network model.

3.2.1. Capacity and routing constraints.

$$\sum_{l\in L}\sum_{d\in D} h_{e,d,d,l}\times c_l \le u_e \qquad \forall e \in E; \tag{2}$$

$$h_{e,d,\,d',\,l} \le h_{e,d,d,l} \qquad \forall e \in E; \forall d, \forall d' \in D; \forall l \in L; \tag{3}$$

$$h_{e,d,d,l} \le m_{e,d} \qquad \forall e \in E; \forall d \in D; \forall l \in L; \tag{4}$$

Constraint (2) restricts the total flow through the edges. This flow may not exceed the capacity of the edge. Constraint (3) imposes that link e can only be used in a virtual path to destination d' for video layer l when e is directly transporting l to a destination node d. Constraint (4) ensures that edge e can only be used to transport (any) layer l directly to destination node d when e is on one of the shortest paths to node d.

3.2.2. Ingoing and outgoing constraints.

$$\sum_{e\in I_v} h_{e,d,d,l} \le 1 \qquad \forall v \in V; \forall d \in D; \forall l \in L; \tag{5}$$

$$\sum_{e\in O_v} h_{e,d,d,l} \le 1 \qquad \forall v \in V; \forall d \in D; \forall l \in L; \tag{6}$$

$$\sum_{e\in I_p}\sum_{d\in D} h_{e,d,d,l} \le 1 \qquad \forall p \in P; \forall l \in L; \tag{7}$$

Constraint (5) guarantees that for every node in the network, there is maximum one incoming edge transporting a specific video layer to a specific destination node. This constraint is necessary to make sure each destination node receives a video layer only once. Constraint (6) imposes that for every node in the network, there is maximum one outgoing edge containing traffic for a specific video layer to a specific destination. Constraint (7) prevents that multiple streams of the same video layer l are incoming on a peer node p. These constraints reduce the solution space of the ILP formulation.

3.2.3. Flow conservation and peer node constraints.

$$\sum_{e\in I_f} h_{e,d,d',l} = \sum_{e\in O_f} h_{e,d,d',l} \qquad \forall f \in F; \forall d, \forall d' \in D; \forall l \in L; \tag{8}$$

$$\sum_{e\in I_p}\sum_{d\in D} h_{e,d,d',l} = \sum_{e\in O_p}\sum_{d\in D} h_{e,d,d',l} \qquad \forall p \in P; \forall d' \in D; \forall l \in L; \tag{9}$$

Constraint (8) guarantees that (direct and indirect) traffic flows through the forwarding nodes in the network (i.e., all nodes except for injector, destination, and peer nodes). Constraint (9) ensures that all incoming direct and indirect flows leave the peer node, where indirect flows can be converted into direct traffic by the peer node.

DOI: 10.1002/dac

3.2.4. Injector and destination node constraints.

$$h_{e,d,d',l} = 0 \qquad \forall e \in I_z; \forall d, \forall d^{'} \in D; \forall l \in L; \tag{10}$$

$$h_{e,d,d',l} = 0 \qquad \forall e \in O_x; \forall x, \forall d, \forall d^{'} \in D; \forall l \in L; \tag{11}$$

$$h_{e,d,d',l} = 0 \qquad e \in I_x; \forall x, d^{'} \in D; \forall d \in D \setminus x; \forall l \in L; \tag{12}$$

$$h_{e,d,d',l} = 0 \qquad e \in I_d; \forall d \in D; \forall d' \in D \setminus d; \forall l \in L; \tag{13}$$

$$(i+1) \times h_{e,d,d,l_i} \leq \sum_{g \in I_d} \sum_{k \in L} h_{g,d,d,k} \qquad \forall d \in D; \forall e \in I_d; \forall l_i \in L; \tag{14}$$

$$\sum_{e \in I_d} h_{e,d,d,l} = \sum_{e \in O_z} \sum_{x \in D} h_{e,x,d,l} \qquad \forall d \in D; \forall l \in L; \tag{15}$$

Constraint (10) ensures that the injector node z only sends data. Because the injector has all layers available, no other constraints are necessary. Constraints (11), (12), and (13) guarantee that all destination nodes only receive data meant for them and prevent destinations from creating data. Constraint (14) ensures that when a destination node d receives video layer l_{i+1}, also, video layer l_i is received on one of d's direct incoming links. Constraint (15) imposes that when a destination node d receives layer l, there must be (at least) a virtual path starting from the injector node z.

3.3. Solving our problem on a tree-video distribution network

To find the optimal solution, the formulation described previously was implemented in the IBM ILOG CPLEX solver (IBM Corporation, Armonk, NY, USA) [28]. To validate our approach, we start with an artificial but analytically tractable sample used case: a video distribution tree. Figure 4 illustrates the tree network topology with the injector node at the root, which inserts the video stream consisting of four video layers into the network. We assume that all video layers have the same constant transport (i.e., bandwidth) cost $c_l = 1$. The injector node is directly connected to a peer node, which on its turn is connected to a level 2 forwarding node. For validation purposes, the bandwidth on this level 2 link is limited and relative to a control parameter α:$\alpha \times 4 \times N$, with N as the number of destination nodes. The level 2 forwarding node is connected to two level 1 forwarding nodes, with each connection having a bandwidth relative to a control parameter β:$\beta \times 2 \times N$. In the scenario that both α and β are set to 1, each destination is able to stream the video at full quality directly from the injector's peer node.

The destination nodes ($N = 10$) are directly connected to a peer node (e.g., a residential gateway), which in turn is connected via an access forwarding node to one of the level 1 forwarding nodes. Each level 1 forwarding node is connected to five access forwarding nodes. The downstream bandwidth on these connections is enough to transport all video layers to each of the destinations. Because the destination's peer node is able to (re) distribute (a part of the) received video layers and is typically located in the access network, the upload bandwidth of these peers is limited (i.e., asymmetrical connection).

The basic network structure of Figure 4 allows us to form an analytical formulation and to calculate the average received video layers on a destination node E:

$$A = \alpha \times |L| \times N \tag{16}$$

$$B = \frac{\beta \times |L| \times N}{fo\_2} \tag{17}$$

DOI: 10.1002/dac

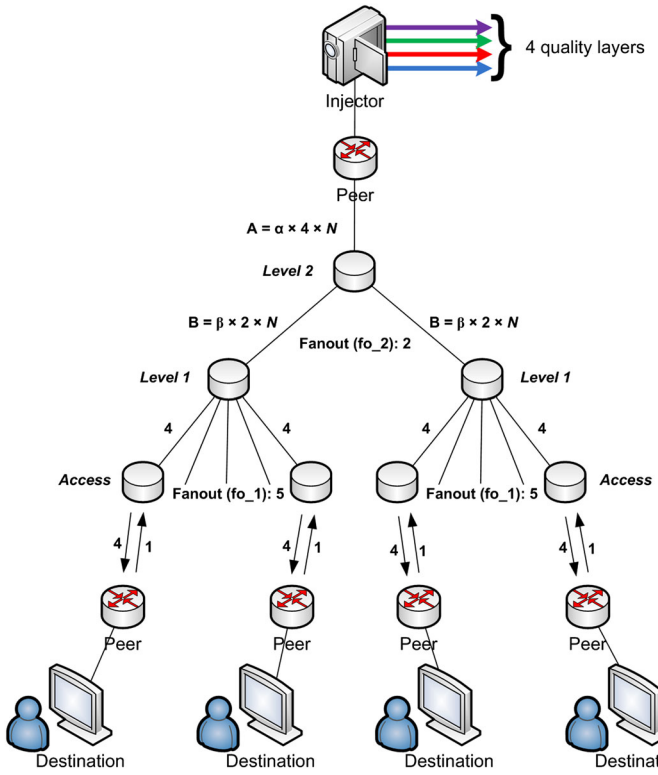

Figure 4. A tree network with the injector node at the root providing a video stream consisting of four quality layers. The injector node is directly connected to a peer node, which on its turn is connected to a level 2 forwarding node. The level 2 forwarding node is connected to two level 1 forwarding nodes. Each level 1 forwarding node is connected to five forwarding nodes that each provides access to one destination's peer node.

$$C = \frac{fo\_2 - \text{peer\_}l1}{fo\_2} \tag{18}$$

$$C' = \frac{\text{peer\_}l1}{fo\_2} \tag{19}$$

$$H = min\left(\frac{A + fo\_1 \times peer\_l1 \times (B - min(|L|, A))}{N - fo\_1 \times peer\_l1}, \frac{B}{fo\_1}\right) \tag{20}$$

$$I = \min\left(|L|,\ A,\ B, C \times (H + up \times peer_a) + C' \times \min(|L|, B)\right) \tag{21}$$

In (16–21 N is the number of destination nodes in the tree network (i.e., $fo\_1 \times fo\_2$), fo_1 and fo_2 are, respectively, the fan-out on level 1 (i.e., 5) and level 2 (i.e., 2), up is the upload bandwidth, $peer\_a$ is the fraction of peer nodes at the destination that are able to distribute received video layers and $peer\_l1$ is the number of level 1 nodes with peer functionality. The average number of received video layers I (equation (21)) is the minimum over four arguments. The first argument of equation (21) states that the average received number of layers is never larger than the number of distinct video layers. The second and third elements (i.e., A defined by equation (16) and B defined by equation (17)) state that the average received number of layers is less than the amount of traffic that the link from the injector's peer to the level 2 node or the link from the level 2 to the level 1 node is able to transport. The fourth argument in equation (21) describes that the highest average quality that can be received is the weighted sum between the part of the tree that has no level 1 peer functionality

DOI: 10.1002/dac

(i.e., denoted by expression C of equation (18)) and the fraction that has level 1 peer functionality (i.e., defined by expression C' of equation (19)). When the level 1 peer exhibits no peer functionality, the average received quality for that sub-tree is not larger than dividing the bandwidth contributed by the injector peer and the level 1 peer, over all destinations that are part of the sub-tree as expressed by equation (20). Because the destination's peer node can contribute to this sub-tree, the fraction of peer nodes times the uplink has to be added. For the destinations that are located under the level 1 nodes having peer functionality, the weighted result is simply limited by the available bandwidth it receives according to expression B (and no more than the number of video layers). Note that in the case that peer_l1 = *fo*_2 (i.e., this generates division by zero), we simply neglect that part because the value of C is 0.

To show the correctness of our proposed model and to study the effect of parameters α and β, a parameter sweep is performed by setting α to 0.2, 0.4, 0.6, 0.8, and 1.0 and β to 0.2, 0.4, and 0.6. The value of $peer\_a = 1.0$, $peer\_l1 = 0.0$, and $k = 1$, because only one shortest-path is possible from a source to a destination. Figure 5 compares the results (represented as dots) of using our model on the tree topology with the analytical solution (depicted by solid lines), when the access link bandwidths are symmetrical (i.e., up and download capacity of four video layers) and asymmetrical (i.e., upload capacity of one and download capacity of four video layers). When symmetrical (access) links are used, each destination receives all video layers if α and β are sufficiently high; however, asymmetrical (access) links require more bandwidth in the core network in order to allow each destination to receive the video in full quality. Note that our ILP solution indeed produces the results predicted by the analytical approach.

Figure 6 depicts the results of the parameter sweep in the situation where asymmetrical access links are used when all, six out of ten or no access peer has uploading capabilities of one layer and none, one, or two out of two level 1 node (s) exhibit peer functionality. Bringing peer functionality into the core network increases the average received quality significantly, even when a large part of the access peer cannot contribute downloaded information to the rest of the network. Again, Figure 6 shows that both the analytical solution and our model produce the same results.

4. USED CASE: EUROPEAN PARLIAMENT STREAMING

In traditional P2P video distributing networks, selection of nodes to download video chunks from (i.e., choking) is performed in a selfish manner. Typically, those nodes are selected that have the highest bandwidths connections [4, 11, 12]. In order to model this, the ILP formulation is extended with a constant parameter $r_{x,y}$ that represents the minimum bandwidth on the shortest-path (i.e., $k = 1$) between nodes x and y (i.e., the link that has the smallest maximum bandwidth u_e is used for $r_{x,y}$).

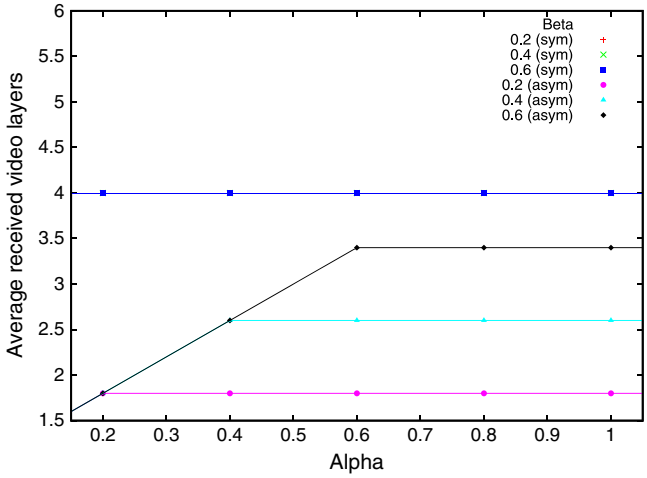

Figure 5. Results of using our model on the tree-based topology (represented as dots) compared to the analytical solution (depicted by solid lines), when using symmetrical versus asymmetrical (access) link bandwidths. In case of symmetrical access links, all results coincide for presented situation.

DOI: 10.1002/dac

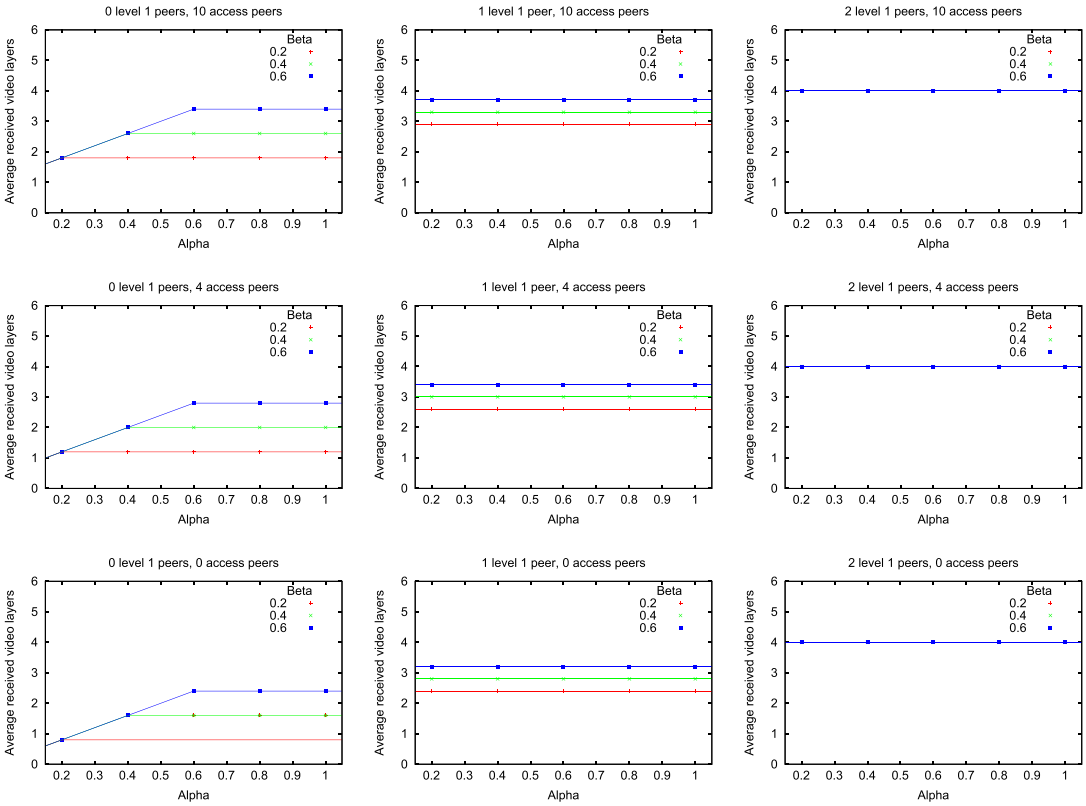

Figure 6. Results of our model compared to the analytical solution, when peer functionality is brought into the core network, when all, 60% or no access peer has uploading capabilities. None, one, or two of the level 1 nodes exhibit peer functionality, and asymmetrical bandwidths are used in the access network.

The objective function to optimize that represents the traditional form of node selection can be formulated as follows:

$$R = \text{maximum}\left(\sum_{s\in\{z\cup P\}}\sum_{e\in O_s}\sum_{d\in D}\sum_{l\in L} h_{e,d,d,l}\times r_{s,d}\right) \tag{22}$$

By optimizing objective function R, peer nodes have the incentive to transport video layers to destinations that seem to have the highest bandwidth connection, which, in principle, is exactly the same as having the destinations selecting (the injector or peer) nodes to download from that have the highest bandwidth connection. Because the injector peer nodes are encouraged by objective function R to send as much video layers as possible, it is important to prevent unnecessary transport of data between nodes, which is taken care of by constraint (equation (7)).

In Figure 7, a ring-of-trees topology is depicted, which is a representative for a core network in Belgium. The injector node is placed on the main ring and γ is the parameter controlling the available bandwidth on the main ring links connecting the four other root nodes. The number of video qualities is (again) set to four layers; the fan-out for the level 2 node is set to 2 and for the level 1 node to 5. The peers on the access level are assumed to be asymmetrical with a download capacity of four layers and an upload capacity of one layer. Because two shortest-paths are possible between a source and a destination, k is set to 2. Parameters α and β are both set to 0.6 (to ensure effects are visible when changing parameter γ), and γ is varied between 0.1 and 1.0.

Figure 8 shows the results of using our strategy, which purpose is to deliver each destination a maximum minimal quality and the traditional solution, where each destination maximizes its

DOI: 10.1002/dac

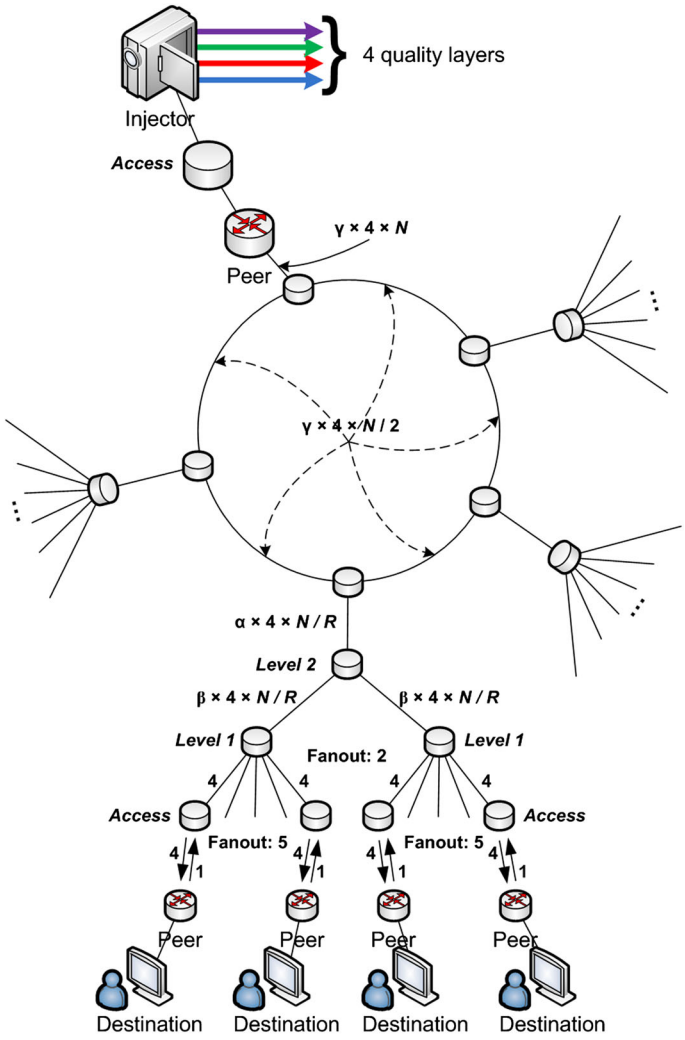

Figure 7. A ring network connecting R forwarding nodes. Each ring node acts as a root for a tree topology, build up in the same way as in Figure 4.

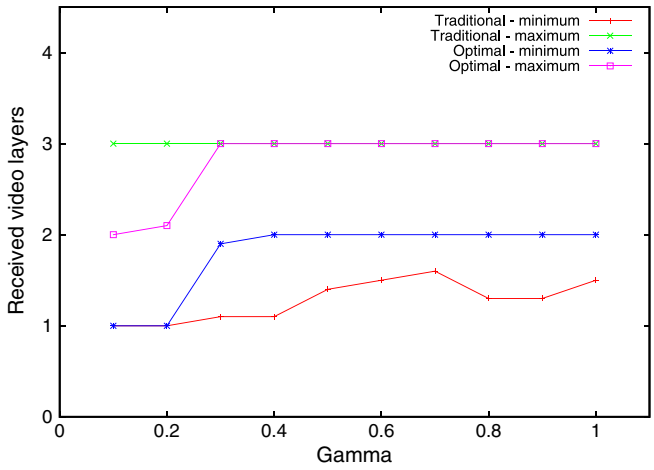

Figure 8. Comparing our strategy with the traditional method on the ring-of-trees network topology. Because both methods produce similar results in terms of average received quality, this figure shows the minimum and maximum number of received video layers (averaged over ten independent simulation runs).

DOI: 10.1002/dac

received quality. In order to obtain results from the ILP solver [28] within reasonable times (i.e., 24 h), we select ten random destination nodes that are allowed to receive the video and the results show the average over ten independent simulation runs. Because of the symmetrical nature of the topology, both methods produce similar results in terms of average received quality by the destinations. However, as Figure 8 illustrates, the gap between the minimum and maximum received quality for our strategy is smaller, indicating that our strategy indeed maximizes the minimum quality that is received by all nodes.

A second study is distributing a live video from the European Parliament. Figure 2 illustrates the topology that we use in our experiments, where the network link bandwidths are expressed in the maximum number of transportable video layers. The (live) video consists of four layers and the injector peer is located in France. We set $k=1$ for the traditional strategy and for our optimal method $k \in \{1,2,3\}$ shortest-paths. Again, in order to obtain results from the ILP solver [28], ten random destination nodes are selected and connected to one of the country's forwarding nodes. All peer nodes that are connected to a destination node have an uplink capacity of one video layer and a download bandwidth of four layers in the case of homogeneous end-devices. To model heterogeneous end-devices, we cap the download capacity of 60% of the peer nodes to one layer, 30% are able to receive two video layers, and 10% are allowed to get the video in its highest quality (i.e., four layers). The results are based on ten independent simulation runs.

Figure 9a illustrates the fraction of the destination nodes receiving a specific number of video layers, when using homogeneous end-devices. When applying the traditional (selfish) method for neighbor/piece selection, most destinations watch the video in its base video layer. Only a few destinations are able to receive the video in two or four video layers. When our optimization strategy is used, compared to the traditional method, a smaller fraction of the destinations acquires the video stream consisting of only one video layer and a much larger part of the destination nodes obtains the video in two video layers. Although no destination node is able to watch the video at full quality using our strategy (i.e., receiving all four video layers), Figure 9a shows that our method offers a more robust solution because a larger fraction of nodes receives layer l_1. In the situation that heterogeneous end-devices are modeled, Figure 9b depicts that fewer nodes are receiving two video layers than in the homogeneous case. However, because a quality cap is enforced into the network, freed bandwidth becomes available and allows our strategy to enforce some nodes to receive three video layers.

5. HEURISTIC METHOD FOR ROUTING OF MULTILAYER VIDEO IN A P2P NETWORK

Because exactly solving the previously described mathematical formulation is only feasible for relatively small network topologies, we propose in this section a heuristic optimization strategy. Our

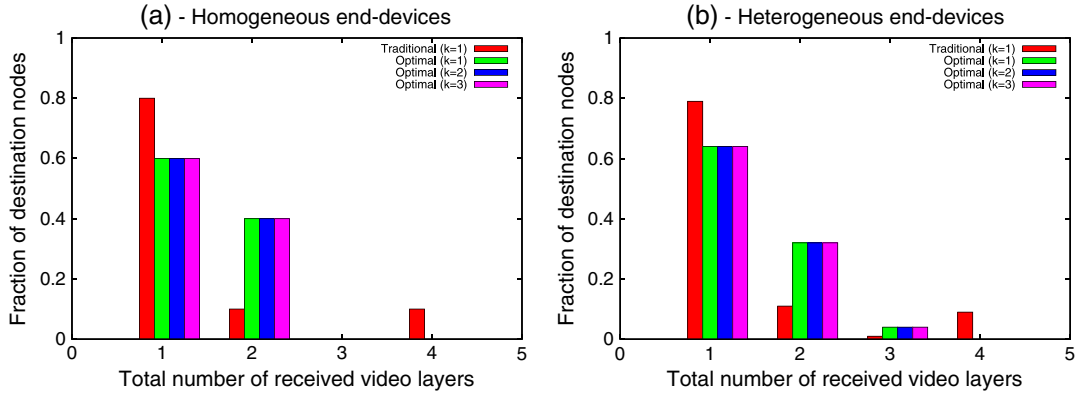

Figure 9. Comparing our optimization strategy with a typical traditional (i.e. $k=1$) Peer-to-Peer video streaming methodology on mesh network topology (i.e. $k \in \{1,2,3\}$); (a) homogeneous end-devices (i.e. each destination is allowed to receive the video feed at its highest quality) and (b) heterogeneous end-devices: 60% of the users are able to stream at maximum one layer, 30% at maximum two layers, and 10% at four video layers.

DOI: 10.1002/dac

heuristic method is based on SA, a stochastic optimization strategy used to find an (close to) optimal solution for a problem. The solution space is randomly sampled and occasionally inferior solution states are accepted in order to jump out of local minima. Before we provide our proposed algorithm based on SA, we explain in section 5.1 the alteration of the h variable used to model the underlay-overlay-routing problem for our heuristic method of multilayer video streaming in a P2P network. Section 5.2 continues with a detailed description of our heuristic strategy while a validation and evaluation of the proposed strategy is provided in section 5.3.

5.1. Problem formulation for distributing multilayer video over a P2P network topology

In this section, e is an edge from E, x a node from set $\{P \cup D\}$, and l is a video layer from list L. The binary variable is altered to:

- $h_{e,x,l}$ is 1 iff link e is used to transport layer l to node x (i.e., a peering or destination node).

The h variables in combination with c_1 allow the calculation of the bandwidth carried by the links and will be used to ensure flow conservation through the network. Figure 10 illustrates an example of transporting layer l to destination node d. The injector node z sends the stream over link e_1 to peering node p. Peer p sends the video layer via link e_3 to forwarding node f, which on its turn forwards l to destination d using link e_5. All other h variables (not depicted in Figure 10) are set to 0.

The combination of the fixed parameters and variables h allows us to describe an underlay-overlay-routing model and is solved using our stochastic heuristic strategy. Compared with the mathematical model, variable h can be simplified because our cleanup process (refer to section 5.2.4) guarantees only feasible routing solutions are considered.

5.2. Stochastic heuristic optimization strategy

Our optimization strategy is inspired by the generic optimization strategy SA, where random states are generated and accepted based on the quality of a new state compared with the current solution state. In order to compare two generated states, equation (23) depicts the sum that expresses a solution into a numeric value, which adheres to our proposed objective strategy; the numeric value of the solution state is the sum of the benefit values b_i according to the received video layers l_i over all destination nodes d in the network topology. When the solution state value is optimized, the minimum received number of video layers at each destination node is maximized.

$$solution\ value = \sum_{l_i \in L} \sum_{d \in D} \sum_{e \in I_d} b_i \times h_{e,d,l_i} \tag{23}$$

In order to jump out of local minima (or maxima), SA uses the Metropolis criterion [29] to calculate the probability of temporarily accepting an inferior proposal.

$$M(delta,\ avg\_delta,\ T) = e^{\frac{-delta}{\frac{-avg\_delta}{\ln(\beta)} \times T}} \tag{24}$$

Parameter *delta* is the numeric difference between two proposals, for example, calculated by the sum in equation (23). New proposals that are slightly worse than the current state have a higher chance to be temporarily accepted. The generated (pseudo) random numbers are uniformly

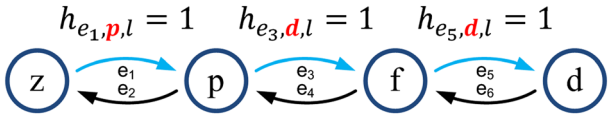

Figure 10. Example indicating the usage of variables h, where the injector node z sends a video layer l destined for node d. The video layer is transported over link e_1 to peering node p. Peer p sends the layer over links e_3 and e_5 to destination node d.

DOI: 10.1002/dac

distributed between 0 and 1. Because the magnitude of *delta* is problem dependent, we generalize the probabilities by using an average value *avg_delta* and parameter β to control the starting probability of the average *delta* value. During the course of the simulation, the chance of accepting a less attractive proposal decreases according to the so-called cooling schedule. The temperature T of the cooling schedule starts with a relatively large value, causing the simulation to start with relatively high chances of accepting a less attractive state (i.e., when $T_0 = 1$, β is the initial probability of accepting the average inferior proposal). By lowering the temperature, the acceptance probability of inferior states decreases over time. In this way, the optimization strategy is able to search a large part of the solution space and finally narrows down to an (almost) optimal solution.

The decision to accept a new proposal (i.e., a newly created route or a solution state) is based on the Metropolis criterion of equation (24). Table II summarizes all input parameters for the optimization method, appended with a brief description of the parameters and the standard value used in our experiments.

5.2.1. Initialization of configuration parameters. The main process of finding the optimal solution is given by the algorithm in Figure 11. First, all configuration parameters are initialized (A.02):

- Average delta value: when constructing new routes and solution states, computed by generating N inferior proposals using a dummy local optimization strategy.
- The start value for the temperature parameter T is set (i.e., T_0).

5.2.2. Starting the temperature steps and the iteration chain. The optimization strategy then continues as long as the temperature parameter T is larger than the predetermined stopping temperature

Table II. List of parameters used by our optimization method

Parameter	Description	Default value
T_0	Starting temperature, used to define the acceptance probability for the Metropolis criterion	1
T_{stop}	Stopping temperature, when the temperature drops below this value, the optimization process finishes	0.001
α	Used by the exponential cooling schedule to define the rate the temperature T decreases in each temperature update	0.99
β	Initial probability of accepting an inferior proposal, where the value of $delta = avg\_delta$	0.9
N	Number of dummy inferior proposals needed to calculate *avg_delta*	100
ICL_1	Iteration chain length: the number of iteration per temperature step	500

```
A.01 findOptimalSolution() {
A.02       initializeParameters();
A.03       while(T > T_stop) {
A.04             for(i = 0 : ICL_1) {
A.05                   initiateTransaction();
A.06                   if(congestedLinks()) {
A.07                         dropVideoLayerFromCongestedLink();
A.08                   }
A.09                   sources = cleanUp();
A.10                   createNewRoute(sources, T);
A.11                   if(acceptNewSolution(T)) {
A.12                         if(bestSolution() && !congestedLinks()) {
A.13                               markNewBestSolution();
A.14                         }
A.15                   } else {
A.16                         performRollBack();
A.17                   }
A.18             }
A.19             T *= α;
A.20       }
A.21       useBestSolution();
A.22 }
```

Figure 11. Main method highlighting the simulated annealing-inspired optimization approach.

DOI: 10.1002/dac

T_{stop}. During each temperature step, an iterative chain is executed that results in the generation of a number of solution states (i.e., ICL_1, the iteration chain length) that are accepted (or rejected) based on the Metropolis criterion. We propose the use of a transaction mechanism for an easy switch back to the current network state in case of a rejection (A.05).

5.2.3. Link dropping strategy. When the network contains congested links (i.e., links that currently require more bandwidth than they have available according to u_e), a random network link e is selected from the set of congested links and a random layer l is dropped, transported to a random node x (i.e., by setting parameter $h_{e,x,l}=0$ and x is a peering or destination node) (A.07). After dropping a layer from a link, a cleanup process is necessary to guarantee that all streams flow correctly in the network.

5.2.4. Cleanup process. A cleanup process is started (A.09) that makes sure that each destination node d from D receives video layers in order (and removing layers that do not fulfill this property), and each peering node p from P is only receiving a video layer when it is actually (re) sending it to another peering or destination node (and vice versa). The cleanup method returns per quality layer a list of nodes that is able to act as a source to another peering or destination node (i.e., injector combined with multiple peering nodes).

5.2.5. Constructing a new download route. To find an optimal solution, we propose an opportunistic method for creating new routes. Even if a new proposal for transporting a video layer to a node crosses already occupied links, we still consider these routes temporarily. By accepting solution states that are infeasible, we allow the optimization strategy to jump out of local minima and to perform a natural way to select links to drop video layers from. The process of starting a new video stream download is explained in Figure 12 and starts with selecting a random destination node (i.e., a destination node that is not yet receiving all video layers) (B.02). The next video layer that the destination currently is not receiving is requested from a random node, can_source, that acts as a candidate source (i.e., injector or one of the peering nodes) (B.03). Next, the shortest-path (measured in network hops) between the final destination and the candidate source node is constructed using Dijkstra's algorithm. When the candidate source is not yet part of the list of available sources (i.e., the list containing the injector and peering nodes that are already receiving the video layer, gained by the cleanup method) (B.05), a random node is selected from the list of already active source nodes (B.06), and the shortest-path between the actual source node and the candidate source node is prepended to the video stream path (B.07). Next, the decision to accept or reject the proposed path is based on the number of full links it uses, which is provided to the Metropolis criterion (B.10). When accepting the new route, line B.11 marks the download of the video layer in the network topology (i.e., by setting $h_{e,x,l}=1$ on each link e on the constructed path for the specific video layer l, where x is the randomly chosen actual or candidate source node or destination node x that is closest to e).

```
B.01 createNewRoute(sources, T) {
B.02       (destination, layer) = selectRandomDestinationAndLayer();
B.03       can_source = selectRandomNode(P ∪ {z});
B.04       path = shortestPath(can_source, destination);
B.05       if(!sources.contains(can_source)) {
B.06            source = selectRandomNode(sources);
B.07            path.prepend(shortestPath(source, can_source));
B.08       }
B.09       full_links = countNumberOfFullLinks(path);
B.10       if(accept(full_links, avg_full_links, T)) {
B.11            insertStream(path, layer);
B.12       }
B.13 }
```

Figure 12. Process of creating a new route from a source node (i.e. injector or peering node) to a destination. The Metropolis algorithm is used to decide whether or not to accept the proposal, based on the number of already occupied links on the shortest-path.

DOI: 10.1002/dac

5.2.6. Accepting or rejecting the new solution state. Figure 13 introduces the process for accepting or rejecting a new solution state. First, the objective value is calculated on line C.02 by using the algorithm specified in equation (23). When the new solution state produces a better objective value, the proposed state is accepted instantly and the current objective value is adjusted (C.05), else the inferior solution state is accepted according to the Metropolis criterion of equation (24). Note that we adjust the objective values to a logarithmic scale to handle the exponential nature of parameter b_i.

5.2.7. Finalizing the temperature steps and the iteration chain. Finally, we determine in line A.11 whether to accept or reject the new solution state (i.e., according to the dropped video layers and the newly created route). In case the proposed state is accepted, the new solution is marked as the best solution (A.13) when the objective value of the new state is better (i.e., larger) than the objective value of the best solution found yet, given that the newly proposed solution state has no congested links (i.e., the proposed solution is practically feasible). When the solution state is rejected, the transaction mechanism is used to roll back to the situation before the proposal (A.16). After performing the predetermined number of steps of the iteration chain, the temperature parameter T is updated for the next iteration process using an exponential cooling schedule (A.19). Logically, the optimization strategy selects the best found solution as the final result (A.21).

5.3. Evaluation of the heuristic approach for multilayer P2P video streaming

To benchmark the optimization heuristic, we compare the output results with measurements of the ILP model implemented with IBM ILOG CPLEX solver (IBM Corporation) [28]. Each topology used in this experiment is the GÉANT network (GÉANT) using the country nodes as forwarding nodes and adding to ten randomly selected ones a < peering and destination > node pair. We assume that the video feed exists out of four distinct layers and our proposed objective strategy is used. In total, we have generated ten network topologies, and Figure 14 compares over all destination nodes, the minimum, average, and maximum received number of video layers for the ILP solver and our optimization strategy. The parameters for our strategy are set according to the values of Table II. The network topologies are ranked according to the average number of video layers a destination node receives. Figure 14 validates our heuristic for the proposed topologies by showing identical results compared with the ILP solver.

6. COMBINING VIDEO LAYER ROUTING AND OPTIMAL PEERING NODE ALLOCATION

Now that we have a heuristic method that is capable to produce (almost) optimal solutions for small topologies, we can scale up the network size (i.e., the number of < peering and destination > node couples in the topology). An interesting question that arises when examining larger network topologies is where to install peering node functionality. Offering peering application intelligence in the network introduces extra costs for network providers, therefore, knowing ideal positions to place the peering nodes can be crucial. To accomplish finding optimal peering node locations,

```
C.01 acceptNewSolution(T) {
C.02       new = solution value, equation (23);
C.03       if(new >= cur ||
C.04          accept(log|D|(cur-new), log|D|(avg_difference), T)) {
C.05             cur = new;
C.06             return true;
C.07       } else {
C.08             return false;
C.09       }
C.10 }
```

Figure 13. Accepting a solution state for the P2P underlay-overlay-routing problem of streaming videos is based on the numeric value of the new state compared with the current solution. When the new proposal is inferior to the current one, the Metropolis criterion is used in the decision to accept the new state.

DOI: 10.1002/dac

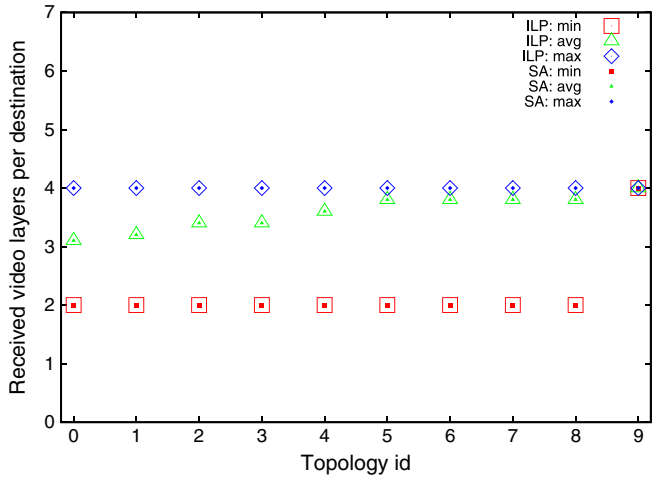

Figure 14. Comparing the number of received video layers of using our optimization heuristic with the exact results measured by the ILP model. Ten distinct topologies are generated, with each ten randomly chosen destination nodes on the GÉANT topology. The topologies are ordered on ascending average received number of video layers, each having a constant bandwidth unit $c_l = 1$.

section 6.1 provides the extension of our optimization strategy. An evaluation of our proposed optimization method is given in section 6.2.

6.1. Extending the optimization strategy to locate ideal positions to place peering nodes

An extra parameter is added to our model to represent the number of allowed peering nodes, *allowed_peers*. The set P now represents forwarding nodes that can be upgraded with peering node functionality. Therefore, the initialization phase (A.02) randomly selects *allowed_peers* peering nodes and marks them as the initial peering nodes (i.e., the remainder of the nodes in P are marked as a forwarding node). An extra iteration chain level is introduced that starts with invoking a method that downgrades a random peering node to a forwarding node and upgrades a random node in P to have peering functionality. The parameter $ICL_2 = 200$ represents the number of times a downgrade and upgrade swap is performed per temperature step. The original inner iteration chain stays untouched and delivers the best objective value that can be reached when using the selected locations for peering nodes. The result of the original iteration chain is then used in the decision to accept or reject the performed swap, similar to algorithm in Figure 13, using the Metropolis criterion (equation (24)). An extra transaction mechanism layer is required to set back the best solution in case of a rejection.

Figure 15 illustrates the transition strategy per distinct video layer for swapping a forwarding node into a peering node (from a to b) and downgrading a peering node into a forwarding node (from c to d). By gracefully changing the node's behavior, we are able to keep most of the previously calculated routes. When a forwarding node is upgraded to exhibit peering functionality, per video layer, only one incoming stream is kept. The binary variable $h_{e,x,l}$ is changed to one for the particular video layer l and all links e on the path of the incoming stream, where x is the id of the new peering node. Limiting a peering node to a forwarding node means for each video layer, replacing variable $h_{e,x,l}$ on the path of the incoming stream, where x is changed into an id of one of the outgoing streams and still satisfying the shortest-path routing principle. All other outgoing streams are removed and to prevent occurring cycles in a route, the chosen outgoing stream is not allowed to be on the link connecting to the node of the incoming link.

6.2. Evaluating peering node placement in a P2P video streaming framework

The network topology that we use is depicted in Figure 2, with the injector node located in France. Each country's forwarding node connects to a destination node via a peering node. The peering node's upload capacity is limited to the average of the maximum incoming bandwidths on the

DOI: 10.1002/dac

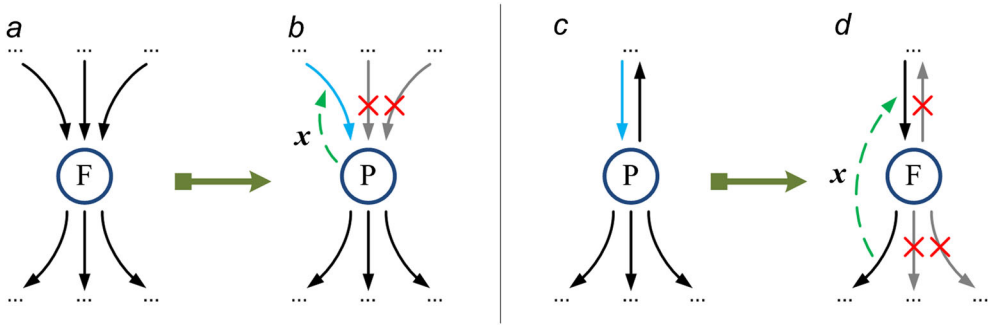

Figure 15. Transition strategy for upgrading a forwarding node (i.e. a peer previously marked as forwarding node) to have peering application functionality is shown from *a* to *b*. Downgrading a peering node to a forwarding node is illustrated from *c* to *d*. In both situations, the binary variable *h* has to be altered for node *x* on an incoming link to represent the new situation correctly.

country's forwarding node. In total, the network topology represents 31 countries, which means one injector, 30 destinations, and at most 31 peering nodes. In order to cancel out noise because of random fluctuations, the results presented in this section are average over ten independent simulation runs. The values for the parameters are chosen as listed in Table II, where $ICL_2 = 200$.

Figure 16 shows the minimum, average, and maximum number of received video layers as a function of the number of *allowed_peers* (i.e., the number of peering nodes exhibiting peering application functionality). As expected, increasing the number of peering nodes in the network increases the average received number of video layers at a destination. Note that the overall network minimum number of received layers is limited by the nodes that only have a download bandwidth of two layers (e.g., Malta: MT). Figure 16 clearly illustrates our proposed strategy: nodes are only allowed to receive a higher video layer when all other nodes are accommodating the lower layers (if possible). Because of the combination of a large number of possibilities to place peering nodes and the relatively small benefit the top layer contributes in the optimization process, fluctuations in the distinct simulation results are observed when the number of allowed peers ranges between 20 and 25.

The bandwidth usage per link in the network is expressed in Figure 17 by measuring the average number of video layers transported by a network link. As already suggested by Figure 16, increasing the number of peering nodes results in an increase in the average bandwidth usage per link. Figure 18 depicts the average number of links transferring a layer divided by the number of destinations receiving the layer. As Figure 18 illustrates, an increase in the number of peering nodes

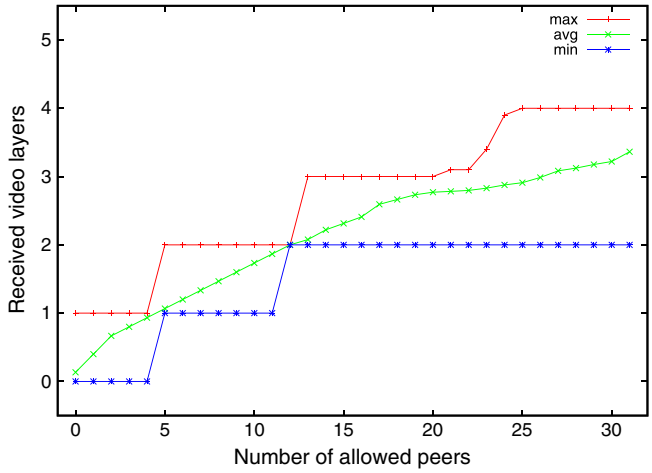

Figure 16. Maximum, average, and minimum number of received video layers as a function of the number of allowed peering nodes in the network. Intuitively, the result of increasing the number of peering nodes is a higher average number of video layers per destination node.

DOI: 10.1002/dac

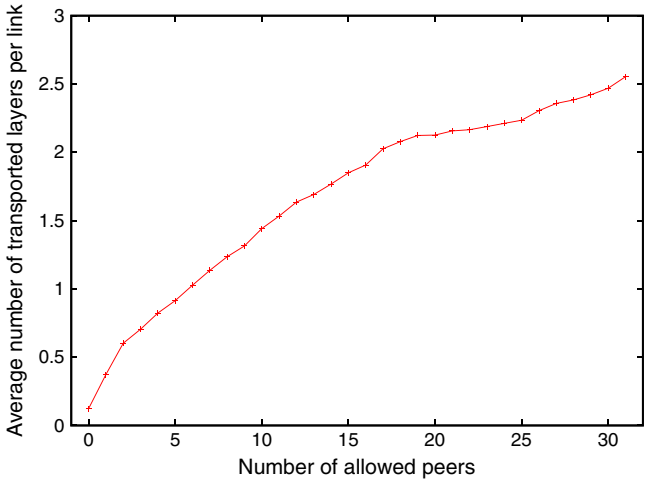

Figure 17. Average number of video layers transported per link in relation to the number of allowed peering nodes. Increasing the number of peering nodes means that the average number of received video layers is increased (refer to Figure 16), causing on average more layers to be transported per link.

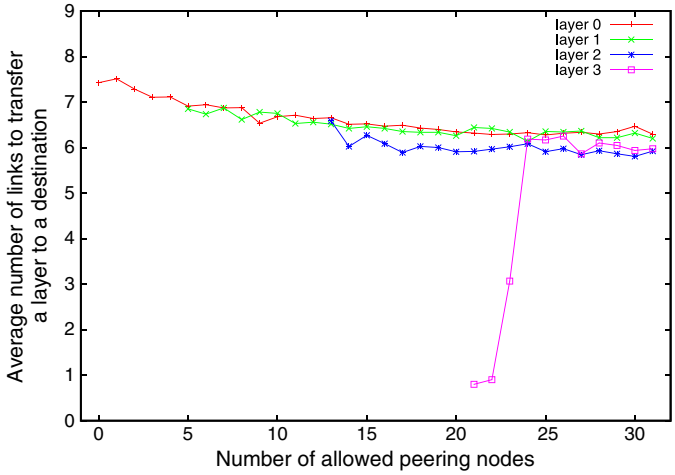

Figure 18. Average number of links transporting a video layer divided by the number of destination nodes receiving the layer, in relation with the number of allowed peers.

results in more efficient usage of the link bandwidths because the average number of links transporting a layer gradually decreases. For our proposed network topology, only five peering nodes are necessary to provide all endpoints the base video layer and 12 peers are required to provide all destinations layer 1. When 31 peering nodes are available, the average number of links to a destination decreases with 9% for the base layer and 5% for layer 1, although no extra endpoints are receiving these video layers. Again, the combination of many options to position the peering nodes and the relatively small benefit layer 3 provides causes small fluctuations in the distinct simulation results when the number of allowed peers is between 20 and 25.

7. CONCLUSION AND FUTURE WORK

Due to increasing bandwidth capacities in the Internet (especially in access networks), next generation P2P video streaming frameworks using multilayer video coding solution provide a good alternative to distribute video feeds. The advantages of using P2P techniques in combination with multilayered video are the ability to easily adapt to heterogeneous end-devices, providing a cost-

DOI: 10.1002/dac

efficient, robust, and scalable solution that naturally exhibits load balancing and reduces server load. From a video service provider's perspective, the objective could be to maximize each destination's minimum video quality (expressed in number of received video layers). The challenge is to optimally use the network infrastructure to transport the video layers to their destinations. To accomplish this, we propose to integrate an orchestrating engine (as an extension to currently existing solutions) that directs the routing process in the overlay network, assuming that this unit has a precise view on the (physical) underlay topology. To study the advantages that this orchestrating unit offers, our paper presents an ILP formulation that is capable to model the underlay-overlay-routing problem of the video layers in the network. By comparing our proposed solution to the strategy applied in traditional P2P video streaming networks, we have shown that the proposed strategy increases the number of nodes that receive the video consisting of more layers than only the base layer. In addition, we propose a stochastic heuristic that is used to find an (close to) optimal solution that can be used by the orchestrating unit, for relatively larger network topologies. We benchmark the heuristic by comparing the results with the results of the exact solver for our evaluation scenario.

To find the optimal solution, the formulation is implemented in an ILP solver [28]. For validation purposes, we have investigated a video distribution structure, for which the optimal solution can be derived analytically. Both the analytical solution and our model produce exactly the same results. When applying both our and the traditional strategy on a ring-of-trees network topology, we show that the difference between the minimum and maximum received video layer is smaller for our solution, providing a more robust solution, without reducing the average received quality. A more realistic used case focuses on distributing a live video feed from the France site of the European Parliament to a selected group of end users (e.g., journalists, translators, …) located in different countries in Europe, where the network topology is based on the GÉANT research network (GÉANT) [6]. We model homogeneous end-devices by allowing each destination node to download the video feed in its full quality. Heterogeneous end-devices have a limited video quality that they can download and we assume that 60% of the end-devices are capable to receive the base layer, 30% can receive two video layers, and only 10% is allowed to get the video in its highest quality. The results show that our optimal strategy significantly increases the fraction of destinations downloading more than one video layer, for both the situation of homogenous and heterogeneous end-devices. However, compared to the traditional method, no destination receives the video in a maximum quality.

Because small hardware investments, by upgrading nodes with peering functionality, can result in major increases in the system's performance (e.g., the number of received layers is increased), an interesting question that arises is which nodes to upgrade. Therefore, we have extended our proposed heuristic optimization strategy to compute along with the video layer routing the best locations to place peering application functionality. The simulation results show that increasing the number of peering nodes in a P2P video streaming network results in an increase in the average received number of video layers at a destination. Consequently, the average bandwidth usage per network link increases because of the increase of the average number of layers an endpoint receives. However, extra peering nodes result in more efficient usage of the network because the average number of links needed to transfer one layer to one destination decreases gradually. This decrease continues even when the number of destinations receiving a particular layer stays the same, that is, a 9% and 5% decrease are seen for, respectively, layer 0 and 1 when no extra endpoints are accommodating these layers.

Alongside upgrading nodes with peering functionality, we plan to integrate advanced caching strategies as proposed by Lin and Lee, and Kao and Lee 20, 30 considering popularity characterizations that are specific to a video [31]. Our techniques can be used to compute the ideal places to implement these proxy caches and can measure the benefits they offer to VSP. The main focus of our study is to optimize the efficiency of (link) bandwidth utilization; however, network management algorithms [32] can be incorporated into our solution to improve quality of experience parameters such as zapping and synchronization time. Security is another important aspect for P2P (live) video streaming networks; additional research is necessary to ensure that our proposed solution supports advanced encryption mechanisms (such as [33]). Because our work focuses on steady-state situations where a video layer is fully obtained from one source peer, the interesting work is to

DOI: 10.1002/dac

expand our model to allow multiple peering nodes to deliver segments of a video layer to a node and to study the effects of nodes (suddenly) leaving or joining the P2P network. Future work involves studying different mechanisms to (further) increase the number of nodes and destinations in the network topology. For instance, we can already lower the number of iterative steps and the stopping temperature to reduce the computation time, however the effects on the solution quality is not studied yet. Another approach to compute the routing of video layers for more realistic scenarios (i.e., larger network topologies) is to use a hybrid solution where our heuristic is used in a multi-step strategy. End users are divided in groups, and in the first step, the video layer routing is determined between these groups. In the next step, our solver can again be used to compute the optimal solution within each (sub) group separately. Of course, solving the video distribution problem in a multi-step way will yield sub-optimal solutions, and the quality of these solutions will depend on the topology at hand.

ACKNOWLEDGEMENTS

The research of Niels Sluijs is funded by a Ph.D. grant of the Agency for Innovation by Science and Technology (IWT). Tim Wauters and Chris Develder acknowledge the Research Foundation – Flanders for its support through a postdoctoral fellowship. Part of this work has been funded by the IBBT OMUS project. The IBBT OMUS project is a project cofunded by IBBT (Interdisciplinary Institute for Technology), a research institute founded by the Flemish Government.

REFERENCES

1. Abbasi U, Mushtaq M, Ahmed T. Delivering scalable video coding using P2P small-world based push-pull mechanism. *Global Information Infrastructure Symposium (GIIS2009)* 2009: 1–7, DOI: 10.1109/GIIS.2009.5307075.
2. Bradai A, Abbasi U, Landa R, Ahmed T. An efficient playout smoothing mechanism for layered streaming in P2P networks. *Peer-to-Peer Networking and Applications* 2014; **7**(2):101–117. DOI: 10.1007/s12083-012-0170-6.
3. Schwarz H, Marpe D, Wiegand T. Overview of the scalable video coding extension of the H.264/AVC standard. *IEEE Transactions on Circuits and Systems for Video Technology* 2007; **17**(9):1103–1120. DOI: 10.1109/TCSVT.2007.905532.
4. Pouwelse JA, Garbacki P, Wang J, Bakker A, Yang J, Iosup A, Epema DHJ, Reinders M, van Steen MR, Sips HJ. TRIBLER: a social-based peer-to-peer system. *Concurrency and Computation: Practice and Experience* 2008; **20**(2):127–138. DOI: 10.1002/cpe.1189.
5. Efthymiopoulos N, Tompros SL, Christakidis A, Koutsopoulos K, Denazis S. Enabling live video streaming services realization in telecommunication networks using P2P technology. *International Journal of Communication Systems* 2011; **24**(10):1354–1374. DOI: 10.1002/dac.1250.
6. GÉANT the pan-European research and education network: http://www.geant.net/Network/NetworkTopology/pages/home.aspx accessed in Jun 2011.
7. Palkopoulou E, Merkle C, Schupke DA, Gruber CG, Kirstädter A. Traffic models for future backbone networks – a service-oriented approach. *European Transactions on Telecommunications* 2011; **22**(4):137–150. DOI: 10.1002/ett.1464.
8. Calvo-Flores MD, Quero JM, Lozano MDR, Miranda AV. Integrating multimedia streaming from heterogeneous sources to JavaME mobile devices. *International Journal of Communication Systems* 2012; **25**(6):763–778. DOI: 10.1002/dac.1251.
9. Deering SE, Cheriton DR. Multicast routing in datagram Internet works and extended LANs. *ACM Transactions on Computer Systems* 1990; **8**(2):85–110. DOI: 10.1145/78952.78953.
10. Bartczak T, Zwierzykowski P. Lightweight PIM – a new multicast routing protocol. *International Journal of Communication Systems* 2014; **27**(10):1441–1458. DOI: 10.1002/dac.2407.
11. Cohen B. Incentives build robustness in BitTorrent. *1st Workshop on Economics of Peer-to-Peer Systems* 2003: 1–5.
12. Mol JJD, Pouwelse JA, Meulpolder M, Epema DHJ, Sips HJ. Give-to-get: free-riding resilient video-on-demand in P2P systems. *Proceedings of SPIE, Multimedia Computing and Networking Conference (MMCN2008)* 2008: 1–8 (article: 681804). DOI: 10.1117/12.774909.
13. Azzedin F. Trust-based taxonomy for free riders in distributed multimedia systems. *International Conference on High Performance Computing and Simulation (HPCS2010)* 2010: 362–369. DOI: 10.1109/HPCS.2010.5547108.
14. Azzedin F.Taxonomy of reputation assessment in peer-to-peer systems and analysis of their data retrieval. *The Knowledge Engineering Review* 2014; **29**(Special Issue 04):463–483. DOI: 10.1017/S0269888914000174.
15. Chatzidrossos I, György D, Fodor V. Server guaranteed cap: an incentive mechanism for maximizing streaming quality in heterogeneous overlays. *Lecture Notes in Computer Science: Networking 2010* 2010; **6091/2010**:315–326. DOI: 10.1007/978-3-642-12963-6_25.

16. Piatek M, Krishnamurthy A, Venkataramani A, Yang R, Zhang D, Jaffe A. Contracts: practical contribution incentives for P2P live streaming. *Proceedings of the 7th USENIX conference on Networked systems design and implementation* 2010: 1–14.
17. Ding Y, Liu J, Wang D, Jiang H. Peer-to-peer video-on-demand with scalable video coding. *Computer Communications* 2010; **33**(14):1589–1597. DOI: 10.1016/j.comcom.2010.04.025.
18. Zhengye L, Yanming S, Ross KW, Panwar SS, Yao W. Layer P2P: using layered video chunks in P2P live streaming. *IEEE Transactions on Multimedia* 2009; **11**(7):1340–1352. DOI: 10.1109/TMM.2009.2030656.
19. Ramzan N, Quacchio E, Zgaljic T, Asioli S, Celetto L, Izquierdo E, Rovati F. Peer-to-peer streaming of scalable video in future Internet applications. *IEEE Communications Magazine* 2011; **49**(3):128–135. DOI: 10.1109/MCOM.2011.5723810.
20. Lin CS, Lee I-T. Applying multiple description coding to enhance the streaming scalability on CDN-P2P network. *International Journal of Communication Systems* 2010; **23**(5):553–568. DOI: 10.1002/dac.1087.
21. Fouda M, Fadlullah Z, Guizani M, Kato N. A novel P2P VoD streaming technique integrating localization and congestion awareness strategies. *Mobile Networks and Applications* 2011: 1–10. DOI: 10.1007/s11036-011-0342-2.
22. Iqbal R, Shirmohammadi S. An analytical approach to model adaptive video streaming and delivery. *Proceedings of the 2010 ACM workshop on Advanced video streaming techniques for peer-to-peer networks and social networking (AVSTP2P2010)* 2010: 55–58. DOI: 10.1145/1877891.1877905.
23. Hu H, Guo Y, Liu Y. Peer-to-peer streaming of layered video: efficiency, fairness and incentive. *IEEE Transactions on Circuits and Systems for Video Technology* 2011; **21**(8):1013–1026. DOI: 10.1109/TCSVT.2011.2129290.
24. Kucharzak M, Walkowiak K. Optimization of flows in level-constrained multiple trees for P2P multicast system. *The Second International Conference on Advances in P2P Systems (AP2PS 2010)* 2010: 106–111.
25. Capone A, Elias J, Martignon F. Routing and resource optimization in service overlay networks. *Computer Networks* 2009; **53**(2):180–190. DOI: 10.1016/j.comnet.2008.09.011.
26. Lin SW, Yu VF, Chou SY. Solving the truck and trailer routing problem based on a simulated annealing heuristic. *Computers & Operations Research* 2009; **36**(5):1683–1692. DOI: 10.1016/j.cor.2008.04.005.
27. Lin SW, Yu VF, Lu CC. A simulated annealing heuristic for the truck and trailer routing problem with time windows. *Expert Systems with Applications* 2011; **38**(12):15244–15252. DOI: 10.1016/j.eswa.2011.05.075.
28. IBM ILOG CPLEX optimizer: http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/ accessed in Nov 2011.
29. Metropolis N, Rosenbluth AW, Rosenbluth MN, Teller AH, Teller E. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics* 1953; **21**(6):1087–1092. DOI: 10.1063/1.1699114.
30. Kao Y-C, Lee C-N. Proxy-assisted P2P and multicast transmission schemes for layered-video streaming over wireless networks. *International Journal of Communication Systems* 2010; **23**(9-10):1167–1188. DOI: 10.1002/dac.1089.
31. Li J, Ma S. Characterization and modeling of video popularity. *International Journal of Communication Systems* 2014; **27**(11):2604–2615. DOI: 10.1002/dac.2493.
32. Lloret J, Garcia M, Atenas M, Canovas A. A QoE management system to improve the IPTV network. *International Journal of Communication Systems* 2011; **24**(1):118–138. DOI: 10.1002/dac.1145.
33. Li J-S, Hsieh C-J, Wang Y-K. Distributed key management scheme for peer-to-peer live streaming services. *International Journal of Communication Systems* 2013; **26**(10):1259–1272. DOI: 10.1002/dac.1394.