Correspondence

An Optimization Framework for QoS-Enabled Adaptive Video Streaming Over OpenFlow Networks

Hilmi E. Egilmez, *Student Member, IEEE*, Seyhan Civanlar, and A. Murat Tekalp, *Fellow, IEEE*

***Abstract*—OpenFlow is a programmable network protocol and associated hardware designed to effectively manage and direct traffic by decoupling control and forwarding layers of routing. This paper presents an analytical framework for optimization of forwarding decisions at the control layer to enable dynamic Quality of Service (QoS) over OpenFlow networks and discusses application of this framework to QoS-enabled streaming of scalable encoded videos with two QoS levels. We pose and solve optimization of dynamic QoS routing as a constrained shortest path problem, where we treat the base layer of scalable encoded video as a level-1 QoS flow, while the enhancement layers can be treated as level-2 QoS or best-effort flows. We provide experimental results which show that the proposed dynamic QoS framework achieves significant improvement in overall quality of streaming of scalable encoded videos under various coding configurations and network congestion scenarios.**

***Index Terms*—Dynamic quality of service, OpenFlow networks, optimization of routing, scalable video, video streaming.**

I. Introduction

Streaming media applications such as WebTV, video-on-demand and videoconferencing require predictable, steady network resources with little delay variation and no packet loss which cannot be always met by the standard best-effort Internet. Therefore, over the past decade, the Internet Engineering Task Force (IETF) has explored several Quality of Service (QoS) architectures, but none has been truly successful and globally implemented. This is because QoS architectures such as IntServ [1] and Diffserv [2] are built on top of current Internet distributed hop-by-hop routing architecture, lacking a broader picture of overall network resources. Although tunneling with MPLS [3] provides a partial solution, it lacks real-time reconfigurability and adaptivity.

OpenFlow is a new programmable network architecture [4] that will allow network service providers to offer new innovative services and network virtualization. One of the new concepts in OpenFlow is decoupling of control and forwarding layers of routing, which, among other benefits, enables an effective means to provide dynamically reconfigurable QoS options. The main contribution of this paper is to propose an optimization framework for dynamic routing of QoS traffic at the OpenFlow's control layer. The proposed framework differs from existing QoS architectures [1], [2] where we offer a new per-flow prioritization scheme that is based on dynamic routing preferences.

Naturally, there will be an extra cost for requesting QoS from the service provider. If the video is encoded at a single layer, such as using H264/MPEG-4 AVC, the entire video needs to be served with either QoS or best-effort. While serving the entire video with QoS clearly provides the highest quality, it comes at a premium cost. On the opposite end, streaming the entire video with best-effort cannot provide any quality guarantee. Alternatively, scalable video coding, such as MPEG-4 SVC [5], [6], encodes video in a base layer and one or more enhancement layers and thereby provides the ability to offer different grades of service at a trade-off point between reasonable guaranteed quality and reasonable cost. In order to guarantee a reasonable quality, it is sufficient to stream the base layer video without any packet loss or delay variation, while enhancement layers can be served with best-effort—or when capacity is available as a QoS stream. Note that all enhancement layers are decoded with reference to the base layer; hence, the need to carry the base layer without packet losses. This paper proposes and evaluates a variety of methods for video streaming with QoS, which trades off video quality performance and cost. Note that we only focus on network-layer adaptive routing for streaming. The adaptive streaming using application-layer rate control is not within the scope of this paper.

There are several notable works in the literature on QoS routing. Masip-Bruin *et al.* [7] present an accurate description of the state-of-the-art and outline research challenges in QoS routing. However, there is no technical problem formulation or solution methodology in their paper. Chen and Nahrstedt [8] provide an overview of QoS routing over next generation high-speed networks. They present different QoS routing problems, their challenges, the QoS routing strategies and an evaluation of existing routing algorithms. Yet, [8] only focuses on connection establishment and resource reservation aspects. Most research on QoS routing deals with solution of multi constrained path (MCP) and constrained shortest path (CSP) problems. To this effect, many approximation and heuristic algorithms have been proposed. Kuipers *et al.* [9] present an overview of constraint-based path selection algorithms for QoS routing. Wang and Crowcroft [10] propose multi-constrained path computing algorithms and also discuss cost metric selections for QoS routing. Xue *et al.*[11] discuss various approximation algorithms for MCP problems and propose an improved approximation scheme. Juttner *et al.* [12] propose a method for delay-constrained least cost QoS routing, which formulates aggregated costs and finds the optimal Lagrange multiplier using the LARAC algorithm. This method runs in polynomial time and produces a theoretical lower bound along with the solution. Chen *et al.* [13] presents two approximation algorithms for the CSP problem, which are alternatives to the LARAC algorithm. These two algorithms achieve smaller average path cost than LARAC, but they run slower. Note that [10]–[13] only consider route calculation task of QoS routing and do not address how to collect information about the state of the network and the available network resources. Moreover, [7]–[9] focus their attention on resource reservation and Intserv-based architectures, and do not discuss either dynamic QoS routing or problem formulation for OpenFlow networks. In [14], Kim *et al.* propose an automatic QoS control over OpenFlow, but it is a completely different approach which is based on rate-shaping. Similar to our framework, there are cross-layer designs for QoS routing over wireless networks (e.g. ad-hoc and sensor networks)[15]–[17], but they cannot be implemented on wired networks. Even though our focus is

Manuscript received August 07, 2011; revised November 04, 2011, March 13, 2012, and May 28, 2012; accepted August 03, 2012. Date of publication December 11, 2012; date of current version March 13, 2013. This work was supported in part by the European FP7 project SARACEN under Grant FP7-248474. An early version of this work is presented at the 18th IEEE International Conference on Image Processing (ICIP 2011), Brussels, Belgium,September2011. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Jiangchuan Liu.

H. E. Egilmez and A. M. Tekalp are with the College of Engineering, Koc University, Istanbul 34450, Turkey (e-mail: hegilmez@ku.edu.tr).

S. Civanlar is with Argela Technologies, Istanbul 34469, Turkey (e-mail: seyhan.civanlar@argela.com.tr).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2012.2232645

on wired networks, there is an initial implementation of wireless extension of OpenFlow (OpenRoads)[18] on which our framework can be implemented with a little effort.

The contributions of this paper are summarized as follows:

- We design a new QoS architecture based on OpenFlow.
- We define a new prioritization scheme based on dynamic QoS routing.
- We propose an optimization framework running on top of OpenFlow's centralized controller paradigm which dynamically fulfills the required end-to-end QoS.
- We apply our methodology to scalable video streaming and evaluate the video quality performance under various coding scenarios.

The rest of this paper is organized as follows: The proposed architecture for QoS-enabled video streaming over OpenFlow networks is introduced in Section II. Section III proposes an optimization framework for OpenFlow controller design, formulating and solving it as a constrained shortest path problem. Simulation experiments and test results are presented in Section IV. Finally, Section V draws some conclusions.

II. An Architecture for QoS-Enabled Adaptive Video Streaming Over OpenFlow Networks

A. Dynamic QoS Support in OpenFlow

In the current Internet architecture, it is not possible to change network routing on a per-flow basis. OpenFlow offers a new paradigm to mainly remedy this deficiency by allowing us to define different routing rules associated with data flows so that the partitions of the network's layout and traffic flows can be instantly modified as in a *Software-Defined Network (SDN)* [19]. The *controller* component of OpenFlow is the brain of the network where routing changes are determined. Thus, different algorithms in the controller associated with different data flows may yield different routing choices. The controller provides access to flow tables, and the rules that tell the network *forwarders* how to direct traffic flows. The proposed controller, depicted in Fig. 1, offers various interfaces and functions, some of which have been part of a router in the classical Internet model. The main interfaces of a controller are:

- **Controller-Forwarder interface**: The controller attaches to a network of forwarders with a secure OpenFlow protocol to send flow tables associated with data flows, to discover network topology, to receive traffic status information and notifications.
- **Controller-Application interface**: The controller provides an open (non-real time), secure interface to application service providers to make reservations of new data partitions (individual or groups of data flows) and even to define new routing rules associated with partitions.

The controller manages several key functions:

- **Topology management**—discovering and maintaining network connectivity topology through data collected or received from forwarders.
- **Route management**determining availability and packet performance of routes to aid the route calculation. This function requires collecting performance data from the forwarders on a periodic or on a notification basis.
- **Route calculation**—calculating and determining routes for different data flows. Several routing algorithms can run in parallel to meet the performance requirements and objectives of different flows. Network topology and route management information are input to this function along with service reservations.
- **Service management**—this is where different data flow request reservations are managed and stored.
- **Service Level Agreement (SLA) management**providing SLA templates to users for different types of allowable data flow patterns.

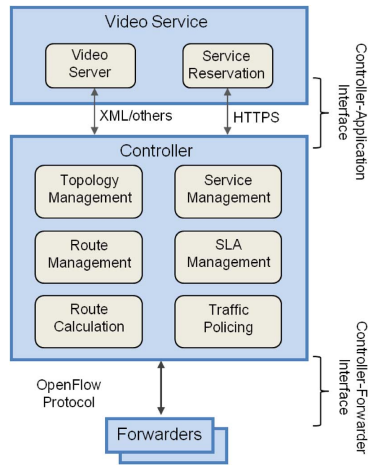

Fig. 1. The OpenFlow controller and interfaces.

- **Traffic policing**—determining if data flows agree with their SLAs, and apply policy rules when they do not.

When a video service (i.e. *service reservation* module) requests a QoS option from the network, it is initially received by the *service management* function of the controller and this function determines if the requested service SLA can be delivered based on other reservations being made. If the reservation is accepted, the network resources are managed for the service starting at the begin time of the data flow. The data flow may start when the video server signals the controller, at which time the *route calculation* function computes the exact route and uploads the flow table to appropriate forwarders. If there is congestion in the network, the *route management* function reactivates the *route calculation* function to determine a new route for the data flow. It is important to note that the *traffic policing* function must also be implemented to make sure the end points conform to their Service Level Agreements (SLAs) stated in their QoS contract.

We can set up different rules for traffic coming from or going to a certain destination (e.g. from a specific server to a client), or of certain type (e.g. video) or protocol (e.g. RTP). The corresponding flow tables are dynamically uploaded to forwarders by the controller. The *route calculation* function determines the QoS routing by optimizing a different cost function other than the hop count. Routes that have larger capacity (even with longer path lengths) may be more preferable to shorter routes that cause packet loss. The QoS flows can be dynamically rerouted based on performance indicators (such as packet loss) collected on the flows' path.

When QoS traffic is placed on a route, more packet loss may be observed on other types of traffic on the shared route. Therefore, any performance optimization process which cares about QoS flows must also consider the impact on best-effort traffic. In order to minimize the adverse effects of QoS provisioning on best-effort flows, we only propose to employ dynamic routing in the proposed QoS architecture and do not exploit resource reservation [1] and/or priority queuing [2] mechanisms. So, we define three different flow types as follows:

- QoS level-1: which will be dynamically rerouted first with highest priority
- QoS level-2: which will be dynamically rerouted after the routes of level-1 traffic are fixed
- Best-effort: with no dynamic rerouting (will follow shortest path)

As illustrated in Fig. 2, the controller generates and sends three sets of flow tables to the forwarders distinguishing the level-1 QoS flow, level-2 QoS flow and the best-effort flow routing for the flows between the same ingress and egress points.

B. QoS-Enabled Scalable Video Streaming Over OpenFlow Networks

In this section, we present methods for QoS-enabled streaming of videos, which are scalable encoded in one base layer and one or more

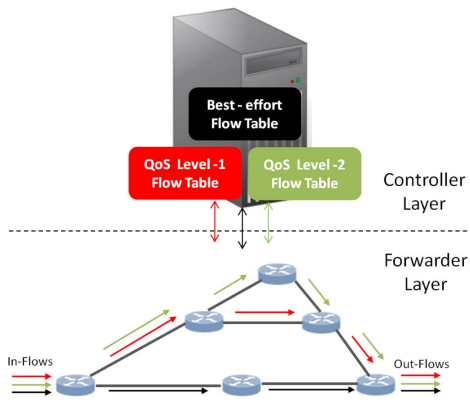

Fig. 2. OpenFlow controller and forwarder interaction with three QoS-levels.

enhancement layers. It is crucial that the base layer is streamed without any packet loss or delay variation to guarantee continuous video playback at the receiving peer, while the enhancement layers may be regarded as discardable, since loss of an enhancement layer packet typically causes only small variation in the received video quality. In the following, we propose two approaches to map the base and enhancement layers of video to level-1 and level-2 QoS, defined in Section II-A, which extends our previous work [20], [21].

Approach-1: The video is MPEG-4 SVC (scalable) encoded, where the base layer packets are sent as level-1 QoS flow, which ensures rerouting to avoid packet losses and minimize delay variation, and the rest of the video (enhancement layer packets) remain as best-effort flows. This approach implies the controller generates two flow tables: one for the QoS flow (level-1); one for the rest of the traffic (best-effort).

Approach-2: The video is MPEG-4 SVC (scalable) encoded, where the base layer packets are sent as level-1 QoS flow, while the rest of the video (enhancement layer packets) are sent as level-2 QoS flow, which provides lower priority rerouting after the routes of level-1 traffic are determined and fixed. This approach implies that the controller generates three flow tables: one for level-1 QoS flow, one for level-2 QoS flow, and one for the best-effort flow for cross traffic. We assume that level-1 QoS will be the most expensive and best-effort will be the least expensive option, and investigate whether rerouting of SVC enhancement layer in $Approach-2$ (more expensive) results in noticeable video quality increase compared to $Approach-1$. We will also evaluate the performances of the proposed *two approaches* against *three benchmarks*:

Benchmark-1: The video is H.264/AVC (not scalable) encoded, and all packets are sent using level-1 QoS support. This approach is used as a benchmark to compare the efficiency of layered coding (MPEG-4 SVC), and represents the highest quality but also the highest cost solution.

Benchmark-2: The video H.264/AVC (not scalable) encoded, and all packets are sent using best-effort flows. This approach represents the least quality but also the least cost solution.

Benchmark-3: The video is SVC encoded, and all packets are sent using best-effort flows.

III. An Optimization Framework for OpenFlow Controller Design

This section presents an optimization framework to design an OpenFlow controller which provides QoS guarantees to designated streams, called QoS flows; that is, to optimize routing dynamically to ensure delivery of QoS flows within specified constraints.

A. Optimization of Dynamic QoS Routing as a Constrained Shortest Path Problem

We pose the dynamic QoS routing as a Constrained Shortest Path (CSP) problem. For the CSP problem, it is crucial to select a cost metric and constraints where they both characterize the network conditions and support QoS requirements. In multimedia applications, the typical QoS indicators are packet loss, delay and delay variation (jitter); therefore we need to determine the cost metric and the constraint accordingly. Obviously, all applications require that the packet loss is minimized for better QoS. However, some QoS indicators may differ depending on the type of the application:

- *Interactive multimedia applications* have strict end-to-end delay requirements (e.g. 150–200 ms for video conferencing). So, the CSP problem constraint should be based on the total delay.
- *Video streaming applications* require steady network conditions for continuous video playout; however, the initial start-up delay may vary from user to user. This implies that the delay variation is required to be bounded, so the CSP problem constraint should be based on the delay variation.

Since in this paper our focus is on video streaming, we will employ delay variation as the constraint in our problem formulation. Note that, it is straightforward to modify the proposed problem formulation for interactive multimedia applications by using the total delay as a constraint instead of delay variation.

In our formulation, a network is represented as a directed simple graph $G(N, A)$, where N is the set of nodes and A is the set of all arcs (also called links), so that arc (i, j) is an ordered pair, which is outgoing from node i and incoming to node j. Let R_{st} be the set of all routes (subsets of A) from source node s to destination node t. For any route $r \in R_{st}$ we define cost $f_C(r)$ and delay variation $f_D(r)$ measures as,

$$f_C(r) = \sum_{(i,j)\in r} c_{ij}, \quad f_D(r) = \sum_{(i,j)\in r} d_{ij} \tag{1}$$

where c_{ij} and d_{ij} are cost and delay variation coefficients for the arc (i, j), respectively. The CSP problem can then be formally stated as finding

$$r^* = \arg\min_r \{f_C(r) | r \in R_{st}, f_D(r) \le D_{max}\} \tag{2}$$

that is, finding a route r which minimizes the cost function $f_C(r)$ subject to the delay variation $f_D(r)$ to be less than or equal to a specified value D_{max}.

We select the cost metric as the weighted sum of packet loss measure and delay variation as follows,

$$c_{ij} = (1-\beta)d_{ij} + \beta p_{ij} \quad \text{for} \quad 0 \le \beta \le 1, \forall (i,j) \in A \tag{3}$$

where p_{ij} denotes the packet loss measure for the traffic on link (i, j), and β is the scale factor. The formula for p_{ij} is as follows,

$$p_{ij} = \begin{cases} \frac{Q^1_{ij}+Q^2_{ij}+T_{ij}-B_{ij}}{Q^1_{ij}+Q^2_{ij}+T_{ij}}, & B_{ij} < Q^1_{ij} + Q^2_{ij} + T_{ij} \\ 0, & B_{ij} \ge Q^1_{ij} + Q^2_{ij} + T_{ij} \end{cases} \tag{4}$$

where B_{ij} is the bandwidth of the link (i, j), Q^1_{ij}, Q^2_{ij} and T_{ij} are the amounts of QoS level-1 traffic, QoS level-2 traffic and best-effort traffic observed on the link (i, j), respectively. It is crucial that forwarders return reasonable estimates of p_{ij} and d_{ij} to determine precise routes. In the proposed controller architecture (Section II), the *route management* function collects data from forwarders (i.e., proper estimates of p_{ij} and d_{ij}) and passes them to the *route calculation* function. At the forwarder level, necessary parameters are estimated as follows:

- Packet loss measure (p_{ij}) is calculated using (4) where B_{ij}, Q^1_{ij}, Q^2_{ij} and T_{ij} are required parameters for the calculation. OpenFlow protocol enables us to monitor the per-flow traffic amounts (i.e., Q^1_{ij}, Q^2_{ij} and T_{ij}) on each link. This is done by per-flow counters maintained in the forwarders. The controller can collect the per-flow statistics on request [22]. The link bandwidth, B_{ij},

```
procedure LARAC(G, s, t, c, d, D_max)
    r_C ← Dijkstra(G, s, t, c)
    if f_D(r_C) ≤ D_max then return r_C
    else r_D ← Dijkstra(G, s, t, d)
        if f_D(r_D) > D_max then return "no feasible solution"
        else
            while true do
                λ ← (f_C(r_C) − f_C(r_D)) / (f_D(r_D) − f_D(r_C))
                r ← Dijkstra(G, s, t, c_λ)
                if f_λ(r) = f_λ(r_C) then return r_D
                else if f_D(r_C) ≤ D_max then r_D ← r
                else r_C ← r
                end if
            end while
        end if
    end if
end procedure
```

Fig. 3. LARAC Algorithm [12].

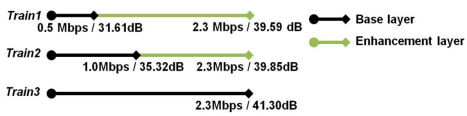

Fig. 4. Rate and quality measures for three different encoding configurations.

can be known by experimenting or setting manually during the network setup.

- Delay is obtained by averaging the observed delay using time stamping (e.g. RTP [23])
- Delay Variation (d_{ij}) is measured as the first derivative (rate of change) of the delay.

The weight β determines the relative importance of the delay variation and the packet losses depending on network and traffic characteristics. For large β, the route selection would be more sensitive to packet losses on the QoS route. Vice versa, for small β the route selection would be more sensitive to delay variation.

In order to find level-1 and level-2 QoS routes, defined in Section II-A, we solve the proposed CSP problem successively two times, first to find the level-1 routes and then to find the level-2 routes by fixing the level-1 routes and modifying the cost parameters accordingly. Note that, the proposed solution method for the CSP problem is presented in the next section (Section III-B). For the QoS level-1 route, r_1, we directly use the estimated packet loss measure (p_{ij}) and delay variation (d_{ij}) parameters to calculate cost coefficients (c_{ij} in (3)) and the CSP problem is solved accordingly to get the optimal route, r_1^*. Then, the QoS level-1 flows (Q_{ij}^1) are rerouted to the optimal route, r_1^*. The QoS level-2 route, r_2, is found after the route of the QoS level-1 flows is fixed. In order to do this, we update the packet loss measure, denoted as p'_{ij}, by removing the observed Q_{ij}^1 traffic from its old route, r_{old}, and placing it on the optimal QoS level-1 route, r_1^*, which can be formulated as follows,

$$p'_{ij}=\begin{cases} \frac{Q^1_{avg}+Q^2_{ij}+T_{ij}-B_{ij}}{Q^1_{avg}+Q^2_{ij}+T_{ij}}, & B_{ij}<Q^1_{avg}+Q^2_{ij}+T_{ij}, (i,j)\in r_1^* \\ 0, & B_{ij}\geq Q^1_{avg}+Q^2_{ij}+T_{ij}, (i,j)\in r_1^* \\ \frac{Q^2_{ij}+T_{ij}-B_{ij}}{Q^2_{ij}+T_{ij}}, & B_{ij}<Q^2_{ij}+T_{ij}, (i,j)\in r_{old} \\ 0, & B_{ij}\geq Q^2_{ij}+T_{ij}, (i,j)\in r_{old} \\ p_{ij}, & \text{otherwise} \end{cases} \tag{5}$$

where Q_{avg}^1 is the average QoS level-1 traffic on the route r_{old}. Then, we recalculate the cost coefficients (c_{ij} in (3)) using the updated packet loss measure (p'_{ij}) for p_{ij} and estimated delay variation (d_{ij}). We solve the CSP problem using the new cost coefficients to get the route, r_2^*, for QoS level-2 (Q_{ij}^2) flows and these flows are rerouted to the route r_2^*.

B. Solution to the Constrained Shortest Path Problem

The solution to the CSP problem stated in (2) will give the minimum cost route satisfying a pre-specified maximum delay variation from source to destination. However, the CSP problem is known to be NP-complete [10], so there are heuristic and approximation algorithms to solve it as proposed in the literature. Here, we propose to use the Lagrangian Relaxation Based Aggregated Cost (LARAC) algorithm [12]. The LARAC algorithm exploits the dual of the CSP problem, which is defined as a maximization problem,

$$\begin{aligned} &\text{maximize} \quad L(\lambda) \\ &\text{subject to} \quad \lambda \geq 0 \end{aligned} \tag{6}$$

where the Lagrangian dual function, $L(\lambda)$, is defined as,

$$\begin{aligned} L(\lambda) &= \min\{f_\lambda(r) - \lambda D_{max} | r \in R_{st}\} \\ &= \{f_\lambda(r) | r \in R_{st}\} - \lambda D_{max} \end{aligned} \tag{7}$$

and

$$f_\lambda(r) = \sum_{(i,j)\in r} (c_{ij} + \lambda d_{ij}) \tag{8}$$

denotes the aggregated cost of a route $r \in R_{st}$ with the Lagrange multiplier $\lambda \geq 0$.

Note that, the minimization term in (7), $\{f_\lambda(r)|r \in R_{st}\}$, can be easily solved by using Dijkstra's shortest path algorithm [24] on aggregated arc costs, for a given λ. The Lagrangian dual function (7) gives the lower bound on optimal value of the original CSP problem for each $\lambda \geq 0$. By maximizing the Lagrangian dual function, $L(\lambda)$, we can find the best lower bound for the optimal solution. This leads us to the solution of dual CSP problem in (6). In order to solve the dual problem given by (6), it is crucial to search for the optimal λ and to determine the criteria for stopping the search. The LARAC algorithm, presented in Fig. 3, provides an efficient search procedure for λ.

In the LARAC algorithm of Fig. 3, $\mathbf{G}$ denotes the network representation as a directed simple graph, $\mathbf{c}$ is the vector of cost coefficients, c_{ij}, for all links, $\mathbf{d}$ is the vector of delay coefficients, d_{ij}, for all links, $\mathbf{c}_\lambda$ is the vector of aggregated link costs coefficients, $c_{ij} + \lambda d_{ij}$, for all links, D_{max} is the maximum tolerable delay, s is the source node, and t is the destination node. Furthermore, $Dijkstra(\mathbf{G}, s, t, \mathbf{c})$, $Dijkstra(\mathbf{G}, s, t, \mathbf{d})$ and $Dijkstra(\mathbf{G}, s, t, \mathbf{c}_\lambda)$ procedures calculate shortest paths using link costs, link delays and aggregated link costs, respectively. The functions $f_C(r)$, $f_D(r)$ (see in (1)) and $f_\lambda(r)$ (see in (8)) give the cost, delay and aggregated cost of route, respectively.

In the first step, LARAC finds the shortest path (route) r_C using link costs. If r_C satisfies the delay constraint, then it is the optimal route. Otherwise, the algorithm checks to determine if a feasible solution exists. To this effect, the algorithm calculates the shortest path (route) r_D using link delays. If the minimum delay route r_D does not satisfy the delay constraint, then a feasible solution does not exist, and the algorithm stops. Otherwise, the algorithm finds the optimal solution by iteratively searching for optimal λ (see while block in Fig. 3). In [12], the authors provide an algebraic approach to efficiently search for λ and to determine the stopping criterion. The corresponding proofs are provided in [25].

It is shown that LARAC is a polynomial-time algorithm that efficiently finds a good route without deviating from the optimal solution [12], [25] in $O([m + nlogn]^2)$ time [26], where n and m are the number of nodes and links, respectively. LARAC also provides a lower bound for the theoretical optimal solution, which leads us to evaluate the quality of the result. By further relaxing the optimality of routes, it provides some flexibility to control the tradeoff between optimality and runtime of the algorithm. Therefore, the LARAC algorithm is well suited for use in real-time dynamic QoS routing.

IV. Results

We have built a comprehensive test network to evaluate the performance of several variants of the proposed QoS routing architecture for scalable video streaming using an open-source network optimization tool, the Library for Efficient Modeling and Optimization in Networks

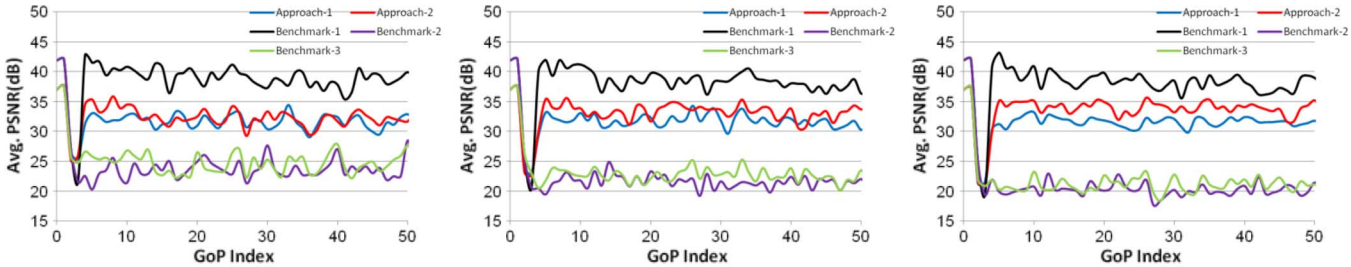

Fig. 5. SNR results obtained by streaming *Train1* over the network with 300 nodes under three different congestion levels: (a) 13, (b) 14, (c) 15.

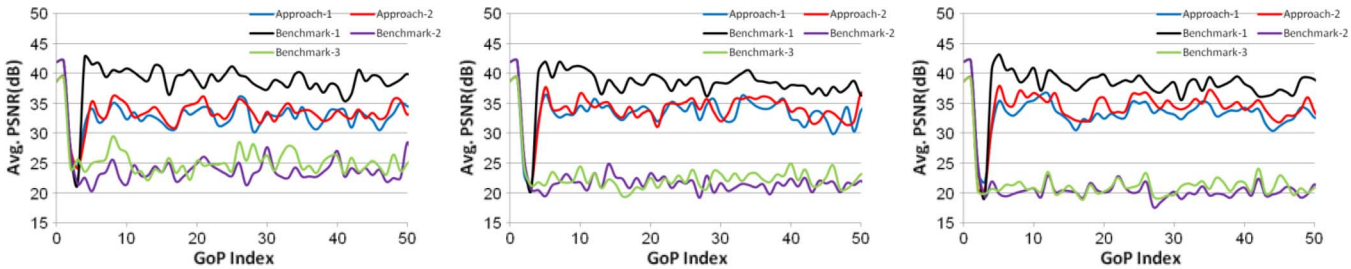

Fig. 6. PSNR results obtained by streaming *Train2* over the network with 300 nodes under three different congestion levels: (a) 13, (b) 14, (c) 15.

TABLE I
PERFORMANCE COMPARISON OF PROPOSED APPROACHES AND BENCHMARKS

Network Size	**Video rate (Mbps)**			**Base=0.5, Enhan.=1.8**			**Base=1.0, Enhan.=1.3**			**Non-scalable=2.3**	
	Method			**A1**	**A2**	**B3**	**A1**	**A2**	**B3**	**B1**	**B2**
300 (nodes)	Congestion Level	*13*	PLR	14%	8%	20%	14%	9%	19%	4%	20%
			PSNR (dB)	31,75	32,40	25,20	32,87	33,49	25,65	38,77	24,32
			Improv. (dB)	7,43	8,08	0,88	8,55	9,16	1,33	14,45	0
		14	PLR	17%	8%	28%	18%	8%	28%	4%	29%
			PSNR (dB)	31,59	32,99	23,17	33,27	33,91	22,89	38,14	22,38
			Improv. (dB)	9,21	10,61	0,79	10,89	11,53	0,51	15,75	0
		15	PLR	20%	8%	37%	20%	7%	35%	4%	36%
			PSNR (dB)	31,40	33,62	21,64	33,20	34,25	21,65	37,93	21,27
			Improv. (dB)	10,12	12,35	0,37	11,93	12,98	0,37	16,65	0

Abbreviations: A1 $= Approach-1$, A2 $= Approach-2$, B1 $= Benchmark-1$, B2 $= Benchmark-2$, B3 $= Benchmark-3$, PLR $=$ Packet Loss Rate, Improv. $=$ Improvement values with respect to $Benchmark-2$.

(LEMON) [27]. We use the Transit-Stub (TS) network model [28], implemented by the Georgia Tech. Internetwork Topology Modeling Tool (GT-ITM) [29]. With GT-ITM, we generate a test network of 300 nodes (with 15 Transit domains). This network is input to the LEMON.

Throughout the simulations, we use the MPEG test video "Train" which is 704 × 576 pixels and 241 frames. We loop original sequence 4 times to generate 900 frames lasting about 30 seconds. This video is scalable encoded using the SVC reference software, Joint Scalable Video Mode (JSVM 9.19), to obtain an SVC base layer stream and an enhancement layer stream where the Group of Picture (GoP) size is set to 16. We set all link capacities to be 15 Mbps and the cross traffic on each link is modeled as an independent Poisson random process which is a good model for the bursty nature of the Internet in sub-second small time scales [30]. The link delays are modeled as Γ-distributed random variables with means 10 ms, 15 ms and 20 ms where we randomly assign these random variables to each link. The maximum tolerable delay variation, D_{max}, is set to 200 ms.

The proposed controller calculates new routes for the video stream by solving the CSP problem posed in Section III-A. To solve this optimization problem, the controller uses the LARAC algorithm (see Section III-B) implemented in LEMON. Dynamic routing is performed at each second which corresponds to 2 GoPs of SVC video approximately. In a nutshell, for each time interval (1 second) the proposed controller method executes following steps:

- detects which video packets are lost on the previous route and calculates the packet loss probability p_{ij} and delay variance d_{ij};
- calculates the cost coefficients c_{ij} using p_{ij} and d_{ij} from previous step for given β value (see (3));
- calculates necessary routes (QoS level-1 and level-2) by solving the CSP problem stated in (2) using the LARAC algorithm;
- reroutes the SVC video layers according to approaches (1 and 2) or benchmarks (1, 2 and 3), discussed in Section II-B.

By matching the lost packets with the Network Access Layer (NAL) units of the SVC video stream, we find which NAL units are lost in the received video stream and erase them. Then, the manipulated video is decoded and the Peak Signal to Noise Ratio (PSNR) is measured.

We execute 9 different test scenarios as combinations of the following simulation parameters:

- *Congestion Level*: We model three congestion levels for shortest path links with Poisson random processes with mean 13 Mbps, 14 Mbps and 15 Mbps, while other links have less congestion (half of the congestion on average).

- *Video encoding*: We encode the "Train" video in 3 different SVC configurations (*Train-1*, *Train-2* and *Train-3*), where base and enhancement layer bitrates and PSNR values are shown in Fig. 4.

We perform Monte-Carlo simulations; that is, we repeat each test scenario 25 times and evaluate the corresponding average PSNR values. For each test scenario, we compare the performance of the traditional best-effort Internet routing and the proposed dynamic QoS routing approaches. The results are depicted in Figs. 5 and 6 in terms of received video quality by calculating the average PSNR values within each GoP, and we observe that the proposed QoS routing approaches ($Approach-1$ and $Approach-2$) significantly outperform the traditional Internet routing ($Benchmark-2$ and $Benchmark-3$). In addition, Table I summarizes the performances of the proposed approaches and benchmarks in terms of the overall average PSNR values. All improvement values are given with respect to $Benchmark-2$.

V. Conclusion

This paper proposes a framework for dynamic rerouting of QoS flows to stream scalable coded videos, consisting of a base and one or more enhancement layers. Inspection of experimental results yields the following observations:

1) The proposed optimization scheme for dynamic rerouting of QoS streams over OpenFlow networks improves the quality of scalable video streaming significantly while causing minimum disturbance on best-effort traffic.
2) Dynamically rerouting the base layer only with QoS ($Approach-1$) attains in the range of 6–12 dB overall PSNR improvement which is significant over sending the entire video stream with best-effort.
3) Dynamically rerouting enhancement layers also ($Approach-2$) provides only marginal improvement (max. 2 dB in overall PSNR) at higher congestion levels given that the base layer bitrate is selected high enough to provide acceptable video quality.
4) As base layer video coding rate increases, both approaches (1 and 2) get closer to rerouting entire non-scalable coded video stream with QoS ($Benchmark-1$), as expected.

Therefore, we conclude that in streaming scalable video over an OpenFlow network, dynamic rerouting of base layer video only ($Approach-1$) is sufficient to attain significant quality improvement over streaming scalable or non-scalable video with best-effort if the network congestion level is low or the base layer's bitrate is high enough to achieve high video quality. Further dynamic rerouting of the enhancement layer ($Approach-2$) pays off the cost of forwarding along QoS level-2 route if the network congestion level is high and base layer's bitrate is low compared to the overall video bitrate. Moreover, since the proposed QoS provisioning scheme is only based on dynamic routing and built on top of OpenFlow's flow reservation paradigm, it has no adverse effect on any other type of flows.

References

[1] R. Braden, D. Clark, and S. Shenker, Integrated Services in the Internet Architecture: An Overview, Internet Engineering Task Force, RFC 1633, Jun. 1994.

[2] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, An Architecture for Differentiated Services, Internet Engineering Task Force, RFC 2475, Dec. 1998.

[3] E. Rosen and Y. Rekhter, BGP/MPLS VPNs, Internet Engineering Task Force, RFC 2547, 1999.

[4] OpenFlow Consortium. [Online]. Available: http://openflowswitch.org.

[5] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the H.264/AVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 9, pp. 1103–1120, Sep. 2007.

[6] H. Sun, A. Vetro, J. Xin, H. Sun, A. Vetro, and J. Xin, "An overview of scalable video streaming," *Wireless Commun. Mobile Comput.*, vol. 7, pp. 159–172, Feb. 2007.

[7] X. Masip-Bruin, M. Yannuzzi, J. Domingo-Pascual, A. Fonte, M. Curado, E. Monteiro, F. Kuipers, P. V. Mieghem, S. Avallone, G. Ventre, P. Aranda-Gutirrez, M. Hollick, R. Steinmetz, L. Iannone, and K. Salamatian, "Research challenges in QoS routing," *Comput. Commun.*, vol. 29, no. 5, pp. 563–581, 2006.

[8] S. Chen and K. Nahrstedt, "An overview of quality of service routing for next-generation high-speed networks: Problems and solutions," *IEEE Netw.*, vol. 12, no. 6, pp. 64–79, Nov./Dec. 1998.

[9] F. Kuipers, P. Van Mieghem, T. Korkmaz, and M. Krunz, "An overview of constraint-based path selection algorithms for QoS routing," *IEEE Commun. Mag.*, vol. 40, no. 12, pp. 50–55, Dec. 2002.

[10] Z. Wang and J. Crowcroft, "Quality-of-service routing for supporting multimedia applications," *IEEE J. Select. Areas Commun.*, vol. 14, no. 7, pp. 1228–1234, Sep. 1996.

[11] G. Xue, W. Zhang, J. Tang, and K. Thulasiraman, "Polynomial time approximation algorithms for multi-constrained QoS routing," *IEEE/ACM Trans. Netw.*, vol. 16, no. 3, pp. 656–669, Jun. 2008.

[12] A. Juttner, B. Szviatovski, I. Mecs, and Z. Rajko, "Lagrange relaxation based method for the QoS routing problem," in *Proc. IEEE INFOCOM*, 2, Apr. 2001, pp. 859–868.

[13] S. Chen, M. Song, and S. Sahni, "Two techniques for fast computation of constrained shortest paths," *IEEE/ACM Trans. Netw.*, vol. 16, no. 1, pp. 105–115, Feb. 2008.

[14] W. Kim, P. Sharma, J. Lee, S. Banerjee, J. Tourrilhes, S.-J. Lee, and P. Yalagandula, "Automated and scalable QoS control for network convergence," in *Proc. INM/WREN'10*, 2010, pp. 1–1.

[15] S. Misra, M. Reisslein, and G. Xue, "A survey of multimedia streaming in wireless sensor networks," *IEEE Commun. Surveys Tutorials*, vol. 10, no. 4, pp. 18–39, Quarter, 2008.

[16] D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris, "A high-throughput path metric for multi-hop wireless routing," *Wireless Netw.*, vol. 11, no. 4, pp. 419–434, Jul. 2005.

[17] Q. Zhang and Y.-Q. Zhang, "Cross-layer design for qos support in multihop wireless networks," *Proc. IEEE*, vol. 96, no. 1, pp. 64–76, Jan. 2008.

[18] K.-K. Yap, M. Kobayashi, R. Sherwood, T.-Y. Huang, M. Chan, N. Handigol, and N. McKeown, "Openroads: Empowering research in mobile networks," *SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 1, pp. 125–126, Jan. 2010.

[19] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling innovation in campus networks," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, 2008.

[20] S. Civanlar, M. Parlakisik, A. M. Tekalp, B. Gorkemli, B. Kaytaz, and E. Onem, "A QoS-enabled OpenFlow environment for scalable video streaming," in *Proc. IEEE GLOBECOM Workshops (GC Workshops)*, Dec. 2010, pp. 351–356.

[21] H. E. Egilmez, B. Gorkemli, A. M. Tekalp, and S. Civanlar, "Scalable video streaming over OpenFlow networks: An optimization framework for QoS routing," in *Proc. IEEE Int. Conf. Image Processing (ICIP)*, Sep. 2011, pp. 2241–2244.

[22] OpenFlow Switch Specification v1.1.0. [Online]. Available: http://www.openflow.org/wp/documents/.

[23] H. Schulzrinne, S. L. Casner, R. Frederick, and V. Jacobson, RTP: A Transport Protocol for Real-Time Applications, Internet Engineering Task Force, RFC 3550, Jul. 2003.

[24] E. W. Dijkstra, "A note on two problems in connection with graphs," *Numer. Math.*, vol. 1, no. 1, pp. 269–271, Dec. 1959.

[25] Y. Xiao, K. Thulasiraman, G. Xue, and A. Juttner, "The constrained shortest path problem: Algorithmic approaches and an algebraic study with generalization," *AKCE J. Graphs. Combin.*, vol. 2, no. 2, pp. 63–86, 2005.

[26] Y. Xiao, K. Thulasiraman, and G. Xue, "Gen-larac: A generalized approach to the constrained shortest path problem under multiple additive constraints," in *Proc. ISAAC*, 2005, pp. 92–105.

[27] LEMON, Library for Efficient Modeling and Optimization in Networks. [Online]. Available: http://lemon.cs.elte.hu.

[28] K. Calvert, M. Doar, and E. Zegura, "Modeling internet topology," *IEEE Commun. Mag.*, vol. 35, no. 6, pp. 160–163, Jun. 1997.

[29] GT-ITM, Georgia Tech Internetwork Topology Models. [Online]. Available: http://www.cc.gatech.edu/projects/gtitm/.

[30] V. Frost and B. Melamed, "Traffic modeling for telecommunications networks," *IEEE Commun. Mag.*, vol. 32, no. 3, pp. 70–81, Mar. 1994.