Multi-stream Frame Rate Guarantee using Cross-layer Synergy

Ahsan Arefin Klara Nahrstedt
University of Illinois at Urbana-Champaign, Urbana, IL
{marefin2, klara}@illinois.edu

***Abstract*—Software Defined Networking (SDN) has provided configurable access to remote network layer components from application hosts during application run-time. We have successfully enabled SDN switches to reduce network resource consumption and improve streaming latency in multi-party 3D tele-immersive (3DTI) applications. Instead of forwarding the same streams to multiple participants from the application host, local network switches replicate network packets towards multiple forwarding paths. However, due to the variable length of application frames generated from the 3DTI application, maintaining constant packet rates at the network switches cannot guarantee required application frame rates towards different forwarding paths. In this poster, we investigate the problem of guaranteeing application frame rates from the network layer switches. We formulate the problem and sketch a scalable solution using OpenFlow, which is a standard protocol developed for SDN.**

I. Introduction

3D tele-immersive (3DTI) system is a distributed platform that connects multiple remote sites containing a large number of input and output devices (e.g., multiple 3D cameras, audio devices and haptic sensors) and creates a virtual shared space for remote interactions. Despite great potential, today's 3DTI still faces significant challenges to maintain desired QoS (quality of service) due to high interactivity requirement and large demand on network and processing loads.

With the advancement of Software Defined Networking (SDN) research over the past few years, network components have become accessible and controllable from application layer [1]. Applications can use this flexibility to design different application features, for example, we have seen the usage of SDN in server load balancing [2], multimedia streaming [3] and network migration [4]. In our current research, we aim to leverage SDN support to dynamically control QoS features in multi-stream and multi-party 3DTI. Our recent work [5] has shown that we can drastically improve network bandwidth usage and end-to-end latency in 3DTI streaming over the Internet by taking SDN support at the edges (i.e., inside local network). We replicate and forward the same streams towards multiple participants from the edge switches rather than from the application hosts, which reduces the length of end-to-end streaming paths and eventually improves network resource consumption and interactive latency.

However, it is still a challenge to maintain different application frame rates towards different forwarding paths using the switch-based multi-party forwarding mechanism. This is because, network packet rates or bit rates at the network switches cannot guarantee required application frame rates. Figure 1 (a) shows that the same network packet rate can result in two different application frame rates. Frames generated from the 3D cameras are usually very large, which form multiple network packets per application frame. Moreover, due to the varying frame size, the number of network packets per frame is not constant during session run-time. Sending frame size information at the switches for each frame is not scalable at run-time. Also, it is hard to identify application frame boundary at the network switches without performing any deep packet inspection, which is computationally very expensive and not feasible for real-time streaming applications.

In this poster, we formulate the problem of multi-rate guarantee first (in Section §II) and then propose a scalable solution (in §II-B). We also analyze theoretical properties of our solution (in §III-A). §IV finally concludes the poster.

978-1-4799-1270-4/13/$31.00 ©2013 IEEE

II. Switch-Based Multi-rate Guarantee

A. Problem Formulation

The switch-based multi-rate forwarding topology for each 3DTI stream can be represented by a tree structure since each participant receives no more than one copy of each stream. Nodes in the tree structure are SDN switches. The application gateway (G) represents end-terminal connected to each tree-node. Gateways are responsible for disseminating streams from local devices and receiving streams from remote participants. The similar representation can be extended for multiple streams.

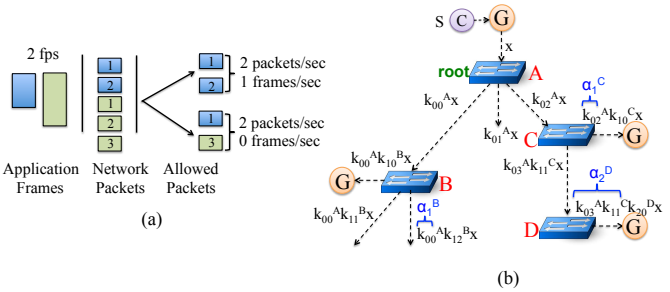

Fig. 1. (a) An example showing the multi-rate guarantee problem, and (b) multi-rate forwarding tree for a single 3DTI stream.

A scenario of multi-rate forwarding for stream S towards 6 remote participants is shown in Figure 1(b). If a switch node Y at level i in this tree structure receives S at application frame rate $\alpha_i^Y x$ form its parent P_Y, the rate for j^{th} forwarding (R_{ij}^Y) out of Y can be defined by $\alpha_i^Y k_{ij}^Y x$, where $k_{ij}^Y \leq 1$ and α_i^Y defines the fraction of the originating frame rate (x) towards Y. The rate towards the local gateway terminal at each participant

is the same as the receiving rate in the local switch, i.e., $k^Y_{i0} = 1$ (except at the root level), since the available bandwidth in local network is usually higher than the available bandwidth over the Internet. At the root level of the tree structure (i.e., at level 0 in Figure 1(b)), the receiving rate is the device originating frame rate (x) and at the leaf nodes the forwarding rates are 0.

Using the above definition, the problem of guaranteeing frame rates over N_Y number of forwarding of a stream from any switch (or tree node) Y at level i in the tree structure can be represented by the following three rules ($0 \leq j \leq N_Y$):

- R^Y_{ij}=$\alpha^Y_i k^Y_{ij} x$ and α^Y_i= $\alpha^{P_Y}_{(i-1)} k^{P_Y}_{(i-1)l}$, where l defines the forwarding path index from P_Y to Y,
- α^Y_0=1 if Y is root, and k^Y_{ij}=0 if Y is a leaf node, and
- For forwarding towards local gateways, k^Y_{i0}=1.

B. Solution Overview

To control application frame rate over a network link, we need to control which frames are allowed to transmit over that link and which frames are not. Moreover, to transmit an application frame, we should allow transmission of all network packets composing that frame. By assigning a single bit in the network packet header for each link in the streaming tree, we can control transmission of network packets (and hence individual frames) over each link. For example, to guarantee application frame rates over 6 forwarding links in Figure 1(b), we can use 6 bits in the packet header. Each bit defines whether the network packets originating from the application frame should be forwarded over the link associated to the bit or not. However, this algorithm is not feasible for two reasons. First, it does not scale, for N forwarding links in the streaming tree, we require N bits in the packet header. Second, any modifications in application frame rate require an update in the packet header which incurs additional overhead in response time.

To solve multi-rate guarantee, we propose forwarding of network packets based on the application layer frame number. The frame number is defined as $(i \mod x)$, where i increases monotonically and x is the maximum achievable frame rate. We can define a list of frame numbers to allow over a link to ensure a specific frame rate. For example, if a 3D video camera generates a stream at the rate of 20 fps (frames per second) (i.e., x=20), to support 15 fps and 10 fps over two forwarding links, the allowed list of frame numbers are {0-2, 4-6, 8-10, 12-14, 16-18)} and $\{0, 2, 4, 6, 8, 10, 12, 14, 16, 18\}$, respectively. We add frame numbers into the packet header using VLAN ID so that network switches can differentiate network packets for each application frame. Finally, we install forwarding rules at each switch node in the tree based on the forwarding list of frame numbers. Thus, any modifications in the frame rate over a remote link only requires update of forwarding rules (to allow different list of frame numbers) at the local switches.

Our solution works in three steps: 1) **Frame Marking**, where we define how individual network packets are marked to matched against the flow table rules inside the network switches, 2) **Frame Relating**, where we define how we can relate frame numbers over any forwarding links to ensure required frame rates, and 3) **Rule Merging**, where we define, how multiple rules for a single stream as well as for multiple streams can be merged in the switch flow table to optimize the number of forwarding rules in the SDN switches.

III. Solution Properties

A. Rate Coverage

Claim: For m bit representation of frame number overriding VLAN ID in network packet header, we can support any integer frame rates in range [1, x], where $x \leq 2^m$.

Proof: Suppose, we want to achieve frame rate r frames per second, where r is an integer between [1, x], and x is the originating frame rate. Suppose d is the highest common divisor between r and x. We can compute p=r/d and q=x/d. Hence, p/q represents an integer fraction, which satisfies $p \leq q$ and $1 \leq p, q \leq x$. Thus by transmitting network packets for p application frames out of each q consecutive frames based on the frame number field, we can guarantee r fps.

B. Flow Rule Optimization

Claim: The maximum number of flow table rules required for forwarding a single stream is bounded by $\lceil \frac{x}{2} \rceil$ towards any number of recipients.

Proof: We can prove this claim easily. For m bits representation of frame number, we can show that the total number of flow table rules is bounded by $\frac{2^m}{2}$ if the size of the forwarding list is lower than $\frac{2^m}{2}$ over a forwarding link. However, if the the size of the list is $\frac{2^m}{2}$+1, there will be at least two rules with single bit difference. We can merge them by using a 'don't care' bit (masking of VLAN ID is allowed in OpenFlow V.1.3). Hence the number of rules cannot exceed $\frac{2^m}{2}$. Similarly we can show that for any $x < 2^m$, the maximum number of unique bit-representation of frame numbers will be bounded by $\lceil \frac{x}{2} \rceil$. Since SDN allows multiple actions on the matching field, the number of rules for a stream does not depend on the number of forwarding links out of a switch.

IV. Conclusion

In this poster, we show that by developing a synergy between application and network layer it is possible to dynamically control application frame rates from the network layer switches. Though we designed the solution for 3DTI applications, we believe that our result will create significant contribution to any streaming applications.

References

[1] Software-Defined Networking, http://bit.ly/LpV2lD, 2012.

[2] N. Handigol, B. Heller, V. Kumar, B. Lantz, and N. McKeown, "Plug-n-server:load balancing web traffic using OpenFlow," in *SIGCOMM*, 2009.

[3] H. Egilmez, B. Gorkemli, A. Tekalp, and S. Civanlar, "Scalable video streaming over OpenFlow networks: An optimization framework for QoS routing," in *ICIP*, 2011.

[4] E. Keller, S. Ghorbani, M. Caesar, and J. Rexford, "Live migration of entire network (and its hosts)," in *HotNets*, 2012.

[5] A. Arefin, R. Rivas, R. Tabassum, and K. Nahrstedt, "OpenSession: SDN-based cross-layer multi-stream management protocol for 3D tele-immersion," *ICNP*, 2013.