Enabling Ultra-Low Delay Teleorchestras using Software Defined Networking*

Extended Abstract†

Emmanouil Lakiotakis
FORTH-ICS
University of Crete
Greece
manoslak@ics.forth.gr

Christos Liaskos
FORTH-ICS
Greece
cliaskos@ics.forth.gr

Xenofontas Dimitropoulos
FORTH-ICS
University of Crete
Greece
fontas@ics.forth.gr

ABSTRACT

Ultra-low delay sensitive applications can afford delay only at the level of msec. An example of this application class are the Networked Music Performance (NMP) systems that describe a live music performance by geographically separate musicians over the Internet. The present work proposes a novel architecture for NMP systems, where the key-innovation is the close collaboration between the network and the application. Using SDN principles, the applications are enabled to adapt their internal audio signal processing, in order to cope with network delay increase. Thus, affordable end-to-end delay is provided to NMP users, even under considerable network congestion.

CCS CONCEPTS

•**Networks** →**Software Defined Networking;** •**Software Defined Networking** →*Routing;* •**Routing** →Latency;

KEYWORDS

Software Defined Networking, Ultra-low delay, Networked Music Performance

ACM Reference format:
Emmanouil Lakiotakis, Christos Liaskos, and Xenofontas Dimitropoulos. 2017. Enabling Ultra-Low Delay Teleorchestras using Software Defined Networking. In *Proceedings of ACM CoNEXT'17 Student Workshop, Seoul/Incheon, South Korea, December 2017 (CoNEXT'17),* 3 pages.
DOI: 10.475/123_4

1 INTRODUCTION

Networked Music Performance (NMP) systems, describe the process where musicians located in different places perform synchronized via the Internet [18]. NMPs belong to ultra-low delay sensitive applications due to the latency requirements they have. In NMP services the maximum affordable delay between the transmitted and the finally reproduced signal should be up to 25 ms. This constraint is denoted as the Ensemble Performance Threshold (EPT) [22].

There are many factors that affect end-to-end delay in NMP systems, which are grouped in two categories: network delay and audio processing delay. Network delay expresses the delay due to data transmission via the network, attributed to physical propagation and the network operating state. On the other hand, the audio processing delay is introduced by the audio capturing, processing and encoding methods, for each transmitter/receiver pair. Due to the delay that audio encoders yield (about 100 ms), they are avoided in NMP services; instead, raw audio is preferred [2].

In this work, our goal is to endow NMP systems with resistance against traffic congestion and link failure cases. To make this feasible, our system introduces collaboration between the NMP application and the network. In more detail, the proposed architecture exploits the flexibility for dynamic network reconfiguration and global view of the network condition that Software Defined Networking (SDN) offers in order to achieve the acceptable latency for NMP [8, 11]. The SDN controller, that orchestrates the network, is responsible for the data path setup that will carry the audio between each transmitter/receiver pair. Additionally, the controller instructs audio flows rerouting in case of congested paths. When the whole network is congested, offering no feasible paths for NMP, the SDN controller communicates with the NMP application, at both the transmitter and receiver sides, in order to switch to another audio configuration set, thus decreasing the audio processing delay and compensating the increased network latency. The proposed architecture, avoids network bandwidth waste, since each node receives a single version of the initial signal, instead of different encoded versions of the same signal [4, 6, 7]. The major difference from similar approaches is that we apply 'smart' routing based on active measurements in the network instead of applying certain policies on the traffic queues of the switches/routers [9, 16, 23, 25], traffic manipulation via TOS matching [1, 24] or using DiffServ approach that is not scalable [27]. We exclusively use end-to-end delay as a rerouting criterion instead of incorporating metrics that reflect the difference between the transmitted and the required by the user audio quality [15, 19].

The main objective of our system is to absorb network delay increase that congestion causes via the blocking delay decrease. The methodology relies on proceeding to low bit-rate transmission and possibly audio quality, while keeping the end-to-end delay under the EPT. When the network recovers from congestion, the NMP application returns to high bit-rate transmission automatically and

*Produces the permission block, and copyright information
†The full version of the author's guide is available as `acmart.pdf` document

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
CoNEXT'17, Seoul/Incheon, South Korea
© 2017 Copyright held by the owner/author(s). 123-4567-24-567/08/06…$15.00
DOI: 10.475/123_4

transparently to NMP users. We evaluated our system in an emulation environment, demonstrating the advantages of the proposed architecture.

2 THE SDN TELEORCHESTRA SYSTEM

As described in Section 1, end to end delay in NMPs is the summary of network and audio processing delay. In mathematical form, end to end delay in NMPs for similar user audio equipment can be modeled as:

$$d_{end-to-end} = 2 \times d_{sound-card} + d_n \quad (1)$$

where $d_{end-to-end}$ represents end-to-end delay, $d_{sound-card}$ shows the delay due to sound-card in transmitter and receiver and d_n expresses the network delay. Delay due to audio processing is alternatively called as blocking delay [6]. Equation (2) describes the blocking delay evaluation:

$$d_{blocking-delay} = \frac{frame\ size}{sampling\ rate} + d_0 \quad (2)$$

In equation (2), frame size denotes the size of audio packets that a user sound-card can process per hardware clock tick, and the sampling rate represents the number of samples the sound-card acquires per second. Finally, d_0 is a constant delay that is due to the sound-card's hardware quality. For blocking delay decrease, the fraction between frame size and sampling rate should be also decreased.

The proposed architecture consists of three components: a SIP, an SDN and a Network Monitoring service as shown in Fig. 1. The SIP service initially measures the audio performance of each user for different audio settings and stores this information. Additionally, it classifies each user as premium or regular based on the privileges that he has. Premium users do not accept low quality transmissions compared with regular users. This category contains musicians that require high quality audio or users (audience) that may have paid for low latency according to SLAs. SIP triggers the application when audio modification is required [5, 20, 26].

The SDN service describes the controller functionality that installs paths for audio transmission. The number of paths towards each user and the path characteristics are defined by the user classification. Each path consists of OVS switches that are instructed by SDN service via the OpenFlow protocol [3]. Also, the SDN service reroutes audio flows when a better path is available. Finally, the Network Monitoring service measures network delay for all paths between users by periodic UDP packet transmission between them. When a new user participates in our system, the SIP service creates an audio profile of the user for various audio parameter combinations. Moreover, it classifies the user as either premium or regular as discussed. Based on the classification results, the SDN service assigns a single path from the audio source to the user and audio transmission starts, and keeps a tunable number of backup paths at the ready. The Network Monitoring service continuously monitors all paths and the SDN service reroutes audio flows to a path that surpasses the active one in performance. If all paths are congested resulting in over-EPT end to end delay, the SIP service contacts the application side in order to switch to an audio mode that has less blocking delay. When the network recovers from congestion, the NMP returns to the previous audio configuration.

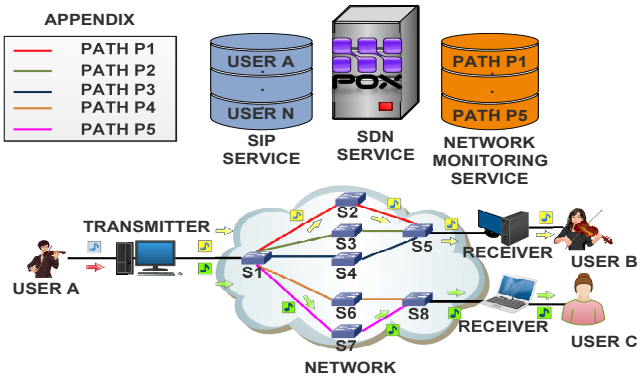

Figure 1: The proposed system architecture.

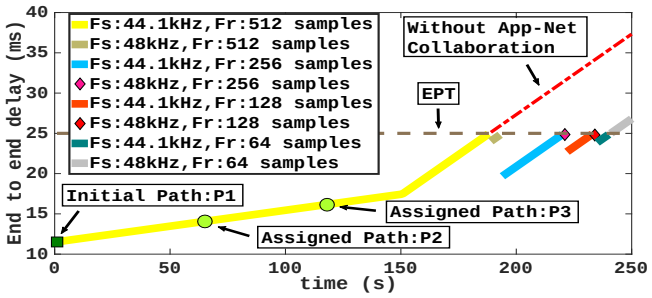

Figure 2: End-to-end delay evaluation.

3 SYSTEM EVALUATION

To evaluate our system, we used real live audio, and Mininet [17] to emulate a realistic network setup. The topology is shown in Fig. 1. POX [12, 21] is selected as the SDN controller. We used Netem traffic control tool in order to increase delay in each path. To avoid redundant reroutes, we used a threshold of at least 2 ms lower latency as the criterion for taking rerouting decisions among available paths. The results of our experimental evaluation are depicted in Fig. 2. F_s denotes sampling rate and F_r the frame size. The proposed Software-Defined NMP solution can be also applied in the real Internet, considering the domains (Autonomous Systems) as "big switches" connected with physical links or classic overlay tunneling mechanisms [10, 13, 14].

Assume User B declares interest for receiving audio from User A. There are three paths between User A and User B in the test topology. At the beginning, the SDN service assigns path P_1 for audio transmission with F_s equal to 44.1 kHz and F_r set to 512 samples. As we increase the network delay in the path, the SDN service reroutes audio flows to paths P_2 (at t = 65 s) and P_3 (at t = 118 s). At t = 189 s, the SIP service informs the NMP to switch to an audio mode with lower blocking delay (F_s equal to 48 KHz and unaltered F_r). Assuming that the network delay increases further, and in order to keep end-to-end delay below EPT, at t = 194 s, the application modifies F_r to 256 samples via interaction with the SIP service. This process takes place during the experimental setup for various audio modes. When the network delay is very high, our system results in best effort delay (after t = 241 s), given the available audio modes and network condition. Comparing our method with the case that application and network could not interact, we achieve delay improvement by 29.6% in end-to-end delay.

REFERENCES

[1] Davide Adami, Lisa Donatini, Stefano Giordano, and Michele Pagano. 2015. A network control application enabling Software-Defined Quality of Service. IEEE, 6074–6079.

[2] D. Akoumianakis, C. Alexandraki, V. Alexiou, C. Anagnostopoulou, A. Eleftheriadis, V. Lalioti, A. Mouchtaris, D. Pavlidi, G. C. Polyzos, P. Tsakalides, G. Xylomenos, and P. Zervas. 2014. The MusiNet project: Towards unraveling the full potential of Networked Music Performance systems. IEEE, 1–6. https://doi.org/10.1109/IISA.2014.6878779

[3] Ian F. Akyildiz, Ahyoung Lee, Pu Wang, Min Luo, and Wu Chou. 2014. A roadmap for traffic engineering in SDN-OpenFlow networks. *Computer Networks* 71 (Oct. 2014), 1–30.

[4] George Baltas and George Xylomenos. 2014. Ultra low delay switching for networked music performance. IEEE, 70–74.

[5] G. Camarillo, R. Kantola, and H. Schulzrinne. 2003. Evaluation of transport protocols for the session initiation protocol. *IEEE Network* 17, 5 (Sept. 2003), 40–46.

[6] Alexander Carôt, Pedro Rebelo, and Alain Renaud. 2007. Networked music performance: State of the art. In *Audio engineering society conference: 30th international conference: intelligent audio environments*. Audio Engineering Society.

[7] Alexander Carôt and Christian Werner. 2007. Network music performance-problems, approaches and perspectives. In *Proceedings of the fiMusic in the Global Villagefi-Conference, Budapest, Hungary*, Vol. 162. 23–10.

[8] Raphael Durner, Andreas Blenk, and Wolfgang Kellerer. 2015. Performance study of dynamic QoS management for OpenFlow-enabled SDN switches. In *Quality of Service (IWQoS), 2015 IEEE 23rd International Symposium on*. IEEE, 177–182.

[9] Hilmi E. Egilmez and A. Murat Tekalp. 2014. Distributed QoS Architectures for Multimedia Streaming Over Software Defined Networks. *IEEE Transactions on Multimedia* 16, 6 (Oct. 2014), 1597–1609.

[10] D. Gkounis, V. Kotronis, C. Liaskos, and X. Dimitropoulos. 2016. On the Interplay of Link-Flooding Attacks and Traffic Engineering. *ACM SIGCOMM Computer Communication Review* 46, 1 (2016), 5–11.

[11] Sergei Gorlatch, Tim Humernbrum, and Frank Glinka. 2014. Improving QoS in real-time internet applications: from best-effort to Software-Defined Networks. IEEE, 189–193. https://doi.org/10.1109/ICCNC.2014.6785329

[12] Sukhveer Kaur, Japinder Singh, and Navtej Singh Ghumman. 2014. Network programmability using POX controller. In *ICCCS International Conference on Communication, Computing & Systems, IEEE*. 138.

[13] Vasileios Kotronis, Xenofontas Dimitropoulos, et al. 2014. Control exchange points: Providing qos-enabled end-to-end services via sdn-based inter-domain routing orchestration. *LINX* 2429, 1093 (2014), 2443.

[14] Vasileios Kotronis, Rowan Klöti, Matthias Rost, Panagiotis Georgopoulos, Bernhard Ager, Stefan Schmid, and Xenofontas Dimitropoulos. 2016. Stitching inter-domain paths over IXPs. In *Proceedings of the Symposium on SDN Research*. ACM, 17.

[15] Harilaos Koumaras, Michail-Alexandros Kourtis, Christos Sakkas, George Xilouris, and Stavros Kolometsos. 2016. In-service Video Quality assessment based on SDN/NFV techniques. IEEE, 1–5.

[16] Himal Kumar, Hassan Habibi Gharakheili, and Vijay Sivaraman. 2013. User control of quality of experience in home networks using SDN. IEEE, 1–6.

[17] Bob Lantz, Brandon Heller, and Nick McKeown. 2010. A network in a laptop: rapid prototyping for software-defined networks. ACM Press, 1–6. https://doi.org/10.1145/1868447.1868466

[18] John Lazzaro and John Wawrzynek. [n. d.]. A case for network musical performance. In *ACM NOSSDAV'01*.

[19] Mu Mu, Matthew Broadbent, Arsham Farshad, Nicholas Hart, David Hutchison, Qiang Ni, and Nicholas Race. 2016. A scalable user fairness model for adaptive video streaming over SDN-assisted future networks. *IEEE Journal on Selected Areas in Communications* 34, 8 (2016), 2168–2184.

[20] Hyunwoo Nam, Kyung-Hwa Kim, Jong Yul Kim, and Henning Schulzrinne. 2014. Towards QoE-aware video streaming using SDN. IEEE, 1317–1322.

[21] L. R. Prete et al. [n. d.]. Simulation in an SDN network scenario using the POX Controller. In *IEEE COLCOM'14*.

[22] Nathan Schuett. 2002. The effects of latency on ensemble performance. *Bachelor Thesis, CCRMA Department of Music, Stanford University* (2002).

[23] Sachin Sharma, Dimitri Staessens, Didier Colle, David Palma, Joao Goncalves, Ricardo Figueiredo, Donal Morris, Mario Pickavet, and Piet Demeester. 2014. Implementing Quality of Service for the Software Defined Networking Enabled Future Internet. IEEE, 49–54.

[24] Sachin Sharma, Dimitri Staessens, Didier Colle, David Palma, Joao Goncalves, Mario Pickavet, Luis Cordeiro, and Piet Demeester. 2014. Demonstrating resilient quality of service in Software Defined Networking. IEEE, 133–134.

[25] Christian Sieber, Andreas Blenk, David Hock, Marc Scheib, Thomas Hohn, Stefan Kohler, and Wolfgang Kellerer. 2015. Network configuration with quality of service abstractions for SDN and legacy networks. IEEE, 1135–1136.

[26] Henry Sinnreich and Alan B. Johnston. 2006. *Internet communications using SIP: delivering VoIP and multimedia services with Session Initiation Protocol* (2nd ed ed.). Wiley Pub, Indianapolis, IN. OCLC: ocm65340953.

[27] Slavica Tomovic, Neeli Prasad, and Igor Radusinovic. 2014. SDN control framework for QoS provisioning. IEEE, 111–114.