Policy-based QoS Management Framework for Software-Defined Networks

Ahmed Al-Jawad, Purav Shah, Orhan Gemikonakli and Ramona Trestian
Faculty of Science and Technology,
Middlesex University, London, UK
E-mails: AA3512@live.mdx.ac.uk, { p.shah, o.gemikonakli, r.trestian}@mdx.ac.uk

***Abstract*—With the emerging trends of virtualization of cloud computing and big data applications, network management has become a challenging problem for optimizing the network state while satisfying the applications' Quality of Service (QoS) requirements. This paper proposes a policy-based management framework over Software-Defined Networks (SDN) for QoS provisioning. The proposed approach monitors the QoS parameters of the active flows and dynamically enforces new decisions on the underlying SDN switches to adapt the network state to the current demanded high-level policies. Moreover, the proposed solution makes use of Neural Networks to identify the violating flows causing the network congestion. Upon detection of a policy violation two route management techniques are implemented, such as: rerouting and rate limiting. The proposed framework was implemented and evaluated within an experimental test-bed setup. The results indicate that the proposed PBNM-based SDN framework enables QoS provisioning and outperforms the default SDN in terms of throughput, packet loss rate and latency.**

Keywords—SDN, Policy-Based Network Management, QoS, Neural Network

I. Introduction

One of the most significant paradigm shifts within the networking industry is represented by the introduction of Software-Defined Networking (SDN). With the traditional networks getting more complex by running the entire control and forwarding planes on the same device, SDN comes to separate these planes and to simplify the communication using the standardized OpenFlow protocol [1]. An SDN controller configures the network elements by distributing the forwarding rules to the switches using low-level language. However, maintaining the network state consistently and establishing individual settings for each network element becomes difficult, especially with the increase in network size. Additionally, the significant growth of video traffic puts pressure on the underlying networks and on the service providers that need to find new solutions to enable efficient resource management while ensuring Quality of Service (QoS) provisioning to their customers and considering Quality of Experience (QoE) as the basis for network control [2]. To overcome these challenges, the Policy-based Network Management (PBNM) proposes a solution to automate the process of network configuration via a set of high-level rules. The combined use of SDN and PBNM brings several benefits compared to legacy networks. SDN reduces the management complexity through the encapsulation of the entire management concept within a central unit. Moreover, the integration of PBNM within SDN could actually enable a simplified management of the data plane through the use of OpenFlow as compared to the complex middleboxes in the traditional networks [3].

The use of PBNM over SDN has been addressed in [4]. The authors present an automatic QoS policy enforcement framework for SDN which monitors the network parameters and adaptively reacts upon the detection of a policy violation. The decision making considers routing management as the main element for policy enforcement. Experimental results show a simple use-case for throughput and loss rate where rerouting of QoS path is considered alone without any rate limiting enforcement. On the other hand, the work in [5] introduces a policy refinement framework over SDN based on logical reasoning. However, the solution did not investigate the techniques for detection and resolution of policy conflicts. In [6] the authors combine the autonomic network mechanisms with the QoS management for SDN. The study presents an extension of OpenFlow and OF-Config protocols to enable dynamic QoS configuration. However, the functionality of the loop between monitoring, enforcement and the violation technique is not clear.

A simple routing algorithm for QoS provisioning over SDN was proposed in [7]. The evaluation was done under a real experimental setup. The work in [8] proposes a compression method to reduce the data traffic on the control path while increasing the network observability. In [9] a probabilistic QoS routing schemes is proposed to select the route that satisfies the given bandwidth constraint.

This paper proposes a policy-based management framework to enable QoS provisioning over SDN-based networks. By using a loop chain approach between network monitoring and policy validation/enforcement, the framework can achieve end-to-end QoS. Moreover, a neural network is used to identify the violating flows causing network congestion and reduce the monitoring overhead. Upon detecting a policy violation the proposed framework implements two route management techniques: rerouting and rate limiting. Experimental test-bed results demonstrate the feasibility of integrating the PBNM architecture over SDN and show that the proposed framework outperforms the default SDN in terms of throughput, packet loss rate and latency.

II. Proposed PBNM-based SDN Framework

Figure 1 illustrates the proposed PBNM-based SDN framework consisting of: **(1) Policy Repository** - stores all the high-level policy rules reflecting the requirements of the agreed service provide-customer services. **(2) Topology Tracker** - maps the physical network elements into a graphical structure and sends the output to the QoS metrics monitoring unit to build a global image of the instantaneous network state. **(3) Admission Control** - accepts or declines network connections based on the availability of network resources. **(4) QoS Metrics Monitor** - measures QoS metrics (e.g., throughput, packet loss rate and delay) by

periodically probing the switches. The constructed view of the network load is used by the violation detector to indicate the misbehaving traffic flows. **(5) Violation Detector** - represents the validation engine to release the necessary measures to converge the network to the state agreed by Service Level Objective (SLO) requirements. **(6) Active Flows Tracker** - tracks the active flow routes and sends a routing table to the monitor unit to estimate the throughput per active flow. **(7) Route Manager** - computes the least loaded path demanded by the application's QoS based on the instantaneous network state and seeks the shortest path for best-effort traffic. **(8) Rate Limiting Manager** - configures the rate limit parameter along the best-effort route.

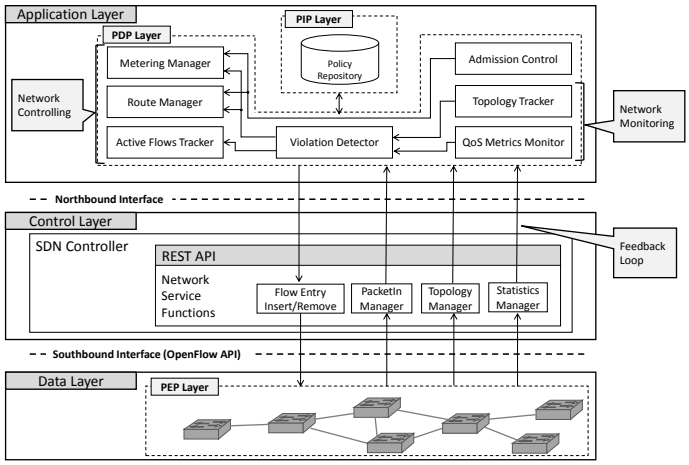

Fig. 1. Proposed PBNM-based SDN Framework

For the purpose of this work, the SLO requirements are defined directly without deriving them from the Service Level Agreement (SLA). The translation and verification between SLA and SLO levels is out of the scope of this work. The framework maps the SLO policies to network policies by manipulating the flow tables of the SDN switches. The SLO policies are stored in an integrated database container.

A. Network Management Function

The functional components of the proposed architecture are mapped to the general three-level PBNM framework (Policy Information Point (PIP) layer, Policy Decision Point (PDP) layer, and Policy Enforcement Point (PEP) layer) [10]. Two cases are identified for managing the network state: (1) *upon receiving a new route request* - Initially, the controller receives a packet-in message and the admission control decides whether to reject or accept the upcoming request based on the resources availability. If the request is accepted, the application type is identified, such that in case of best-effort request, the shortest path is determined using a method based on Dijkstra's algorithm. Whereas in case of QoS application, the least congested path is chosen. (2) *a policy violation is detected* - This case is illustrated in Fig. 2. Initially, the network monitoring component collects periodically flow statistics from the switches. These are used to calculate the available bandwidth and packet loss rate of all the flows, enabling the controller to build a global view of the network load. The violation detector determines whether a high-level policy rule is broken or not. If the policy is broken, it identifies the flow that causes the violation by comparing the measured quality metrics against the high-level policy. This is further used to identify the congested link along the misbehaving flow that causes the violation. Upon a violation, the violation detector either triggers the route manager to choose an alternate route or the rate limiting manager to reduce the bandwidth budget for the background violating best-effort flows.

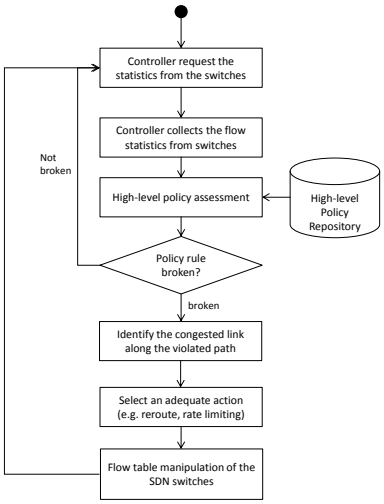

Fig. 2. Network management work flow

B. Identifying Violating Flows using NN

Based on the measured loss rate of the flows, the violation detector can identify the QoS flow that breaks the QoS policy rule. However, it cannot determine the flows that caused the actual congestion. Generally, to enable this, the controller explicitly pulls the statistics from the involved switches as an additional step. Our proposed solution avoids this explicit request and it attempts to predict which flows are involved in the bottleneck. To achieve this, we use the feed-forward Neural Network (NN) for classifying whether the flow is involved in causing the congestion or not. We refer to it as the *violating flow*.

In order to avoid increasing the complexity of the problem, we make use of NN with one hidden layer to identify whether a flow is causing the congestion or not. Higher order of hidden layer has been investigated and the results led to the same outcome as with the one layer NN.

Using the NN as a classifier or predictor involves two stages: (1) *the training stage* where the training data set is used to adjust the weights along the node interconnection, and (2) *the testing stage* where the data set is used to validate the constructed model. A 3-layer NN is used as a classifier with one input layer, one hidden layer of 3 neurons and one output layer. The output y is computed as a sigmoid function f of the weighted sum of inputs x:

$$y = f\left(\sum_{i=1} w_i \cdot x_i\right) \tag{1}$$

where w indicates the weights of the link between the input layer and next network layer, which are estimated during the training phase. The NN has two inputs: the QoS flow throughput and the throughput of the flows sharing the resources. On the other hand, the NN has two outputs indicating the classification classes: violating and non-violating.

In the training phase, the network is trained using an offline learning phase and previously collected data. Generally, offline based learning contributes no overhead to the system as the training is happening outside the normal framework operation. Here, the observations of generated training data set represent the base for building the network model in advance. Thus, the supervised training set is built by generating two types of traffic (e.g., QoS traffic and background traffic representing violating flows) while monitoring the packet loss.

III. Experimental Setup

This section presents the experimental setup, the performance metrics and the evaluation scenarios considered.

A. Experimental Setup

The proposed PBNM-based SDN framework was implemented and tested under the experimental setup illustrated in Fig. 3. The testbed consists of three main elements: (i) Mininet [11] - used to emulate the SDN data plane; (ii) external Floodlight OpenFlow controller [12] - provides RESTful API and network services like the flow entry update; and (iii) the PBNM application layer (described in Section II) - containing the decision making for QoS policy configurations. The SDN controller and the entire PBNM application run on a computer and they are connected via a physical Ethernet link to other computer hosting Mininet. Ofsoftswitch13 [13] is used as a software SDN switch.

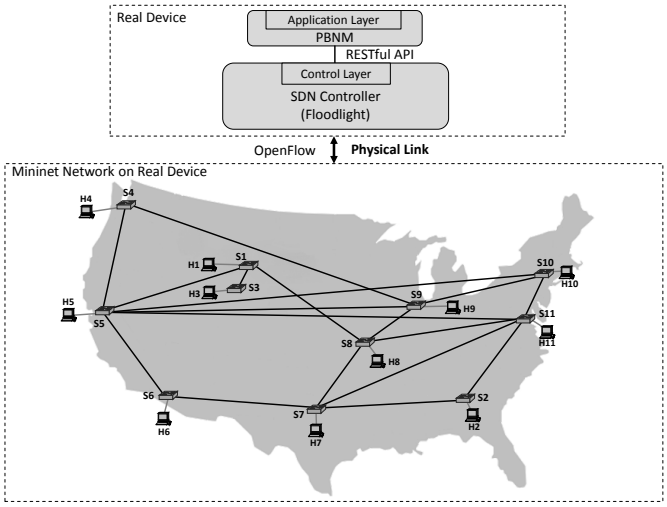

Fig. 3. Experimental Setup using the Sprint Network Topology

To evaluate our approach, a realistic Internet Service Provider (ISP) topology is used. The Sprint IP backbone and customer topology as depicted in Fig. 3 were used for the experimental setup, with the network nodes being replaced by SDN-Openflow enabled switches. The topology was taken from Internet zoo topology [14] and consists of 11 ISP nodes that are located geographically in multiple distinct cities in the US and interconnected through 18 connectivity links. The number of nodes and links, the diameter, and the average outgoing node-degree indicate that the topology is a good candidate for the analysis of our approach. However, because of the processing capacity limitations of the experimental setup, each link in the topology operates at the rate of 1 Mb/s. This does not affect the evaluation performance and the approach can be scaled up to a larger network. Each switch has a host directly connected that generates data traffic.

TABLE I
Parameters of traffic modeling and setup

Parameters	Value
Video bit-rate	563 kbps
Video frame rate	24 fps
Video duration for QoS traffic	10 minutes
Video duration for best-effort traffic	2 minutes
Experiment duration	30 minutes
Traffic mix	Video = 80% HTTP = 20%

Video streaming traffic is generated using the VLC player, while background traffic like HTTP is generated using Ostinato [15] traffic generator tool. In this way, it is possible to evaluate different traffic mix and load on the network. The traffic generated within the experimental setup consists of: guaranteed traffic such as video streaming and best-effort traffic represented by video and web flows used as background traffic. Table I illustrates the video traffic parameters, the experiment duration as well as the traffic mix. The traffic mix ratio is determined based on the statistics provided by Cisco [16] such that 80% of the total traffic is represented by video traffic and the remaining 20% is represented by HTTP traffic. The parameters for the HTTP traffic model [17] used are listed in Table II. The HTTP traffic is modeled as ON/OFF period, where the ON period corresponding to the transmission time and the OFF period corresponding to the packet inter-arrival time. For each traffic request, the source and destination host pairs are selected randomly following a uniform distribution.

TABLE II
Model parameters of web traffic

Parameters	Best-fit Distribution	Mean & Std. Deviation
Main object size	Truncated Lognormal	Mean = 10710 bytes Std. dev. = 25932 bytes
Embedded object size	Truncated Lognormal	Mean = 7758 bytes Std. dev. = 126168 bytes
Number of embedded objects per page	Truncated Pareto	Mean = 5.64 Max. = 53
Reading time	Exponential	Mean = 30 sec
Parsing time	Exponential	Mean = 0.13 sec

The following SLO policy is defined for the case-study: *the QoS policy defines that all flows directed from the source to the destination should receive a defined minimum bandwidth and minimal packet loss rate.*

The performance of the proposed framework is assessed in terms of user perceived QoE for video streaming, using well-known objective metrics, such as: Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity (SSIM) [18]. A mapping of PSNR and SSIM to the nominal Mean Opinion Score (MOS) is given in Table III [19]. MOS is a five point scale used to subjectively assess the users' QoE.

TABLE III
PSNR and SSIM to MOS Mapping [19]

MOS	PSNR	SSIM
5 (Excellent)	≥ 45	≥ 0.99
4 (Good)	≥ 33 & < 45	≥ 0.95 & < 0.99
3 (Fair)	≥ 27.4 & < 33	≥ 0.88 & < 0.95
2 (Poor)	≥ 18.7 & < 27.4	≥ 0.5 & < 0.88
1 (Bad)	< 18.7	< 0.5

B. Evaluation Scenarios

To evaluate the proposed PBNM-based SDN framework under dynamic network conditions and policy violations, a scenario with a mix of QoS and best-effort flows is considered. The proposed solution integrates two mechanisms that could be triggered to overcome the policy violation: rerouting and rate limiting. The QoS policy rule for this scenario is defined as: the QoS video traffic from source Host 2 (H2), see Fig. 3, directed to the destination Host 4 (H4) has a minimum bandwidth threshold of 600 Kb/s and the maximum packet loss rate threshold set to 2%. The characteristics of the QoS video traffic are listed in Table I. The host pair (H2 to H4) was selected to represent the longest distance (number of hops) within the network to increase the likelihood of disturbing the QoS video flow by other background traffic. To disturb the QoS video flow, a mix of video and HTTP traffic as best-effort is generated between random hosts maintaining a 80% to 20% ratio [16].

The monitoring update interval for the proposed PBNM-based SDN framework was set to 3 seconds. This is because the SDN controller is unable to maintain a complete global knowledge of the whole network per single iteration. Using this setup, a first experiment was executed on 20 individual random trials to evaluate the classification provided by the NN model. For each trail, the setup generates new values for the random seed. The identification of violating link achieves an overall accuracy of 98.7% with the following results: true positive rate with 83.3%, true negative rate with 99.4%, false positive rate with 0.01%, and false negative rate with 0.11%. As the result shows, the classifier has correspondingly lower false positive and false negative rates.

IV. Results and Discussions

The performance evaluation compares the performance of the proposed PBNM-based SDN framework against the default configuration of the SDN-based network without the PBNM framework. The default SDN computes the shortest path for all traffic types. The comparison is performed on the same random seed to reproduce a deterministic trail. Each experiment is repeated three times and the average outcomes are evaluated. Both approaches, such as rerouting and rate limiting of the proposed PBNM-based SDN framework are considered. The performance evaluation is done in terms of Throughput, Packet Loss Rate, Latency, PSNR, SSIM and MOS of the QoS video flow.

A. PBNM-based SDN framework with rerouting

In this setup the proposed PBNM-based SDN framework has the rerouting module enabled. Thus, when QoS policy violation is detected the framework will reroute the disturbing traffic and gives priority to the QoS video flow. As a first step in the route setup phase, the route manager selects the least loaded path (S2-S11-S5-S4) for the QoS video traffic between H2 and H4. Figure 4 illustrates the throughput, packet loss rate and latency measurements of the QoS video flow for the PBNM-based SDN framework with rerouting and the default SDN. It can be noticed that three policy violations were detected by the framework.

The results show how the policy condition on the shared link S11-S5 is being strictly violated for the first violation. During the experimental run, the monitoring component identifies at time-stamp 64 that the packet loss rate exceeded the limit of 2% imposed by the QoS policy rule. The packet loss rate is caused due to the shared resources on the common link which becomes congested. Due to this, the violation detector identifies the best-effort flow from Host 11 (H11) to Host 5 (H5) as a disturbing flow and it routes it on an alternative path S11-S10-S5. To determine the violating flow causing the problem, the violation detector uses the supervised neural network to check if the given link is involved. As a consequence, the violation detector releases the event of policy constraint breaching and notifies the route manager. Other violations are identified in the time-stamps 132 and 252.

Generally, rerouting alone would not be a sufficient measure to handle the policy violation problem. For instance, when multiple links carrying several QoS traffic flows with high bandwidth consumption are involved in congestion then it becomes very difficult to balance the network load.

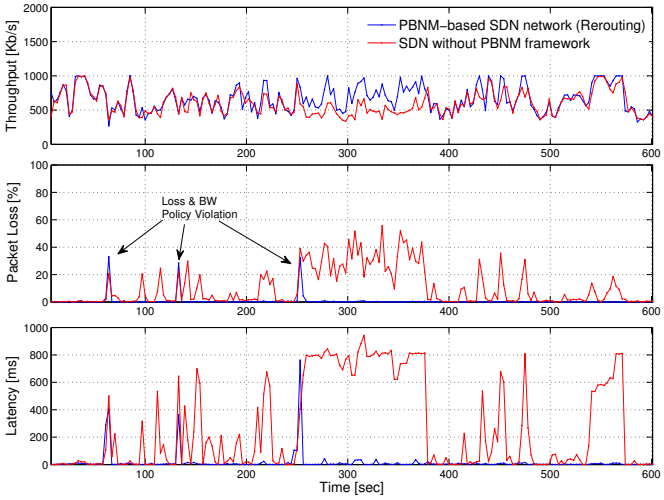

Fig. 4. Throughput, Packet Loss, and Latency of QoS video flow for PBNM-based SDN framework with rerouting and default SDN

Table IV lists the average PSNR and SSIM for the QoS video flow as well as the mapping to the MOS done according to Table III, of both the proposed PBNM-based SDN framework with rerouting and the default SDN. The results indicate that when using the PBNM-based SDN with rerouting, the user perceives the video quality as *Excellent* based on both PSNR and SSIM to MOS mapping. Whereas in the case of default SDN, the user perceived quality for the QoS video flow is *Poor* (PSNR to MOS mapping) towards *Fair* (SSIM to MOS mapping). Thus, by using the proposed PBNM-based SDN framework with rerouting there is an increase of 94% in PSNR as compared to the default SDN.

TABLE IV
Average PSNR to MOS and SSIM to MOS Mapping

	PSNR	MOS	SSIM	MOS
PBNM with rerouting	46.61	5 (Excellent)	0.99	5 (Excellent)
Default SDN	23.97	2 (Poor)	0.94	3 (Fair)

Figure 5 illustrates a comparison snapshot of the QoS video frame from the original transmitted video, the video frame received after the proposed PBNM-based SDN framework performed the rerouting and the video frame as

received using the default SDN. It can be noticed that the QoS video frame quality becomes noticeably poorer relative to the original video frame when the default SDN framework is used with a PSNR of 15.39dB indicating a *Bad* user perceived quality. However, by enabling the proposed PBNM-based SDN framework with rerouting the quality of the video frame improves considerably, with a PSNR of 50.52dB representing *Excellent* user perceived quality.

a) original video frame b) PBNM-based SDN rerouting c) default SDN

Fig. 5. Quantitative video frame quality comparison: a) original image, b) proposed PBNM-based SDN framework with rerouting (PSNR = 50.52dB, MOS=5 -Excellent), and c) default SDN (PSNR = 15.39dB, MOS=1 -Bad)

B. PBNM-based SDN framework with rate limiting

In this setup the proposed PBNM-based SDN framework has the rate limiting module enabled. When a QoS policy violation is detected, the rate limiting module will throttle the output rate of the background best-effort traffic by dropping packets while the traffic flows maintain the same route. This is done, to ensure an end-to-end QoS guarantee for the QoS video flow and to control the high throughput aggregates in the network.

Figure 6 illustrates the throughput, packet loss rate and latency measurements of the QoS video flow for the PBNM-based SDN framework with rate limiting and the default SDN. In this case, the rate limiting manager reduces the data rate to resolve the misbehavior of the background best-effort traffic flows. The results show that if no network adjustment would be considered (e.g., default SDN), the QoS video flow throughput would continue to suffer from the impact of packet loss and delay.

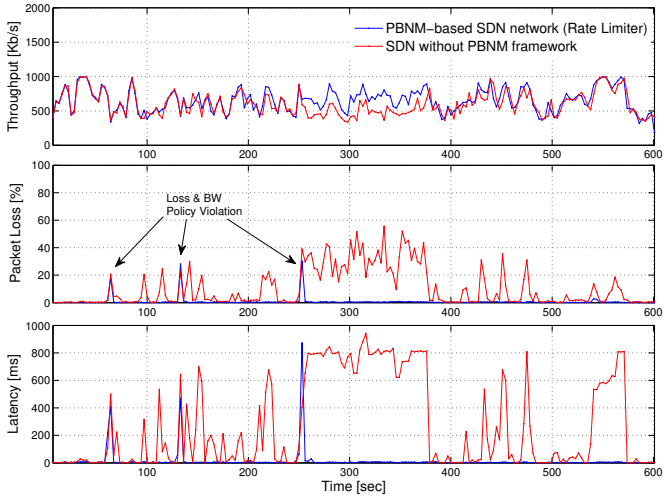

Fig. 6. Throughput, Packet Loss, and Latency of QoS video flow for PBNM-based SDN framework with rate limiting and default SDN

Table V lists the average PSNR and SSIM for the QoS video flow as well as the mapping to MOS, of both the proposed PBNM-based SDN framework with rate limiting and the default SDN. The results are similar to the case where the PBNM-based SDN framework with the rerouting approach is used. It is observed that both methods enable user perceived quality improvements when compared to the default SDN approach. Results show that the proposed PBNM-based SDN framework with rate limiting can achieve up to 91% increase in PSNR with a *Excellent* user perceived quality compared to the default SDN where the user perceived quality is *Poor* (PSNR to MOS mapping) towards *Fair* (SSIM to MOS mapping).

TABLE V
AVERAGE PSNR TO MOS AND SSIM TO MOS MAPPING

	PSNR	MOS	SSIM	MOS
PBNM with rate limiting	45.81	5 (Excellent)	0.99	5 (Excellent)
Default SDN	23.97	2 (Poor)	0.94	3 (Fair)

Figure 7 illustrates a comparison snapshot of the QoS video frame from the original transmitted video, the video frame received after the proposed PBNM-based SDN framework performed the rate limiting and the video frame as received using the default SDN. Similarly to the previous rerouting setup, it can be noticed that the QoS video frame quality is significantly improved by using the proposed PBNM-based SDN framework with rate limiting from *Bad* quality as perceived with the default SDN to *Excellent* quality.

a) original video frame b) PBNM-based SDN rate limiting c) default SDN

Fig. 7. Quantitative video frame quality comparison: a) original image, b) proposed PBNM-based SDN framework with rate limiting (PSNR = 50.45dB, MOS= 5 - Excellent), and c) default SDN (PSNR = 15.39dB, MOS= 1 - Bad)

C. Monitoring Overhead vs. Application Performance

This section analyzes the trade-off between the monitoring overhead introduced and the application performance. Several experimental runs were conducted using the same setup but with different monitoring update intervals: 3, 6 and 9 seconds. The choice of these values, starting from 3 seconds above is done due to the SDN controller that needs time to perceive a consistent image of the entire network and to take the necessary measures to avoid the aftermath of policy violation. The results are listed in Table VI for the default SDN and the proposed PBNM-based SDN framework with rerouting and with rate limiting. It can be seen that as the monitoring update interval increases the application performance decreases. This is because the SDN controller will take longer to detect and respond to the misbehaving best-effort traffic that affects the quality of the QoS video flow. However, even with the increased monitoring update interval, both methods of the proposed

TABLE VI
AVERAGED PERFORMANCE EVALUATION FOR DIFFERENT MONITORING UPDATE INTERVALS (3, 6, AND 9 SECONDS)

Performance Metrics	Default SDN	PBNM with rerouting			PBNM with rate limiting		
		3	6	9	3	6	9
Throughput [Kb/s]	605	645	651	648	621	616	630
Packet Loss [%]	10.22	0.65	1.02	1.35	0.69	0.97	1.38
Latency [ms]	268.67	14.87	13.06	12.15	14.04	13.60	12.43
PSNR [dB]/MOS	23.97/2(Poor)	46.61/5(Excellent)	45.13/5(Excellent)	43.46/4(Good)	45.81/5(Excellent)	44.47/4(Good)	43.22/4(Good)
SSIM/MOS	0.94/3(Fair)	0.99/5(Excellent)	0.99/5(Excellent)	0.99/5(Excellent)	0.99/5(Excellent)	0.99/5(Excellent)	0.99/5(Excellent)

PBNM-based SDN framework outperform the default SDN. For example, for an monitoring update interval of 9 seconds the quality of the QoS video flow is still perceived as *Good* (PSNR to MOS mapping) towards *Excellent* (SSIM to MOS mapping) for both proposed approaches, compared to *Poor* (PSNR to MOS mapping) towards *Fair* (SSIM to MOS mapping) as perceived when the default SDN is used.

Figure 8 shows the overall amount of monitoring overhead introduced on the control path. The results are for different monitoring update intervals regardless of the approach being used, such as rerouting or rate limiting. The results show that the monitoring overhead is inversely proportional to the update interval. For example, the communication overhead is reduced by up to 63% when the update interval changes from 3 to 9 seconds. However, this comes at the cost of twice the packet loss rate and 10% decrease in PSNR. Thus, the trade-off between the introduced overhead and the application performance needs to be considered.

Although the introduction of PBNM scheme in SDN network adds more network overhead than the default SDN, the results show that the performance of the QoS application is significantly improved.

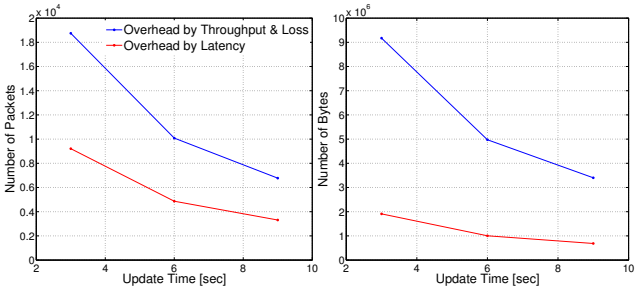

Fig. 8. Monitoring Overhead for 3, 6, and 9 sec. update intervals

V. CONCLUSIONS AND FUTURE WORKS

This paper proposes a policy-based network management framework over SDN for QoS provisioning. The proposed framework makes use of Neural Networks to identify the violating flows causing congestion. Two route management approaches are investigated: rerouting and rate limiting. Experimental results show that the proposed framework outperforms the default SDN in terms of throughput, packet loss rate and latency. The proposed PBNM-based SDN framework with rerouting can achieve up to 94% increase in the average PSNR when compared to the default SDN, increasing the user perceived quality from *Poor* to *Excellent*. As future work, the PBNM-based SDN framework will be extended to integrate a proactive intelligent decision-making system for minimizing the number of SLO violations.

REFERENCES

[1] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[2] R. Trestian, I. S. Comsa, and M. F. Tuysuz, "Seamless multimedia delivery within a heterogeneous wireless networks environment: Are we there yet?" *IEEE Communications Surveys Tutorials*, pp. 1–1, 2018.

[3] A. Lara, A. Kolasani, and B. Ramamurthy, "Simplifying network management using software defined networking and openflow," in *Advanced Networks and Telecommuncations Systems (ANTS), 2012 IEEE International Conference on*. IEEE, 2012, pp. 24–29.

[4] M. F. Bari, S. R. Chowdhury, R. Ahmed, and R. Boutaba, "PolicyCop: an autonomic QoS policy enforcement framework for software defined networks," *Future Networks and Services (SDN4FNS), 2013 IEEE SDN for*, pp. 1–7, 2013.

[5] C. C. Machado, J. A. Wickboldt, L. Z. Granville, and A. Schaeffer-Filho, "Policy authoring for software-defined networking management," in *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*. IEEE, 2015, pp. 216–224.

[6] W. Wang, Y. Tian, X. Gong, Q. Qi, and Y. Hu, "Software defined autonomic QoS model for future Internet," *Journal of Systems and Software*, vol. 110, pp. 122–135, 2015.

[7] A. Kucminski, A. Al-Jawad, P. Shah, and R. Trestian, "Qos-based routing over software defined networks," in *Broadband Multimedia Systems and Broadcasting (BMSB), 2017 IEEE International Symposium on*. IEEE, 2017, pp. 1–6.

[8] A. Al-Jawad, P. Shah, O. Gemikonakli, and R. Trestian, "Compression-based technique for sdn using sparse-representation dictionary," in *Network Operations and Management Symposium (NOMS), 2016 IEEE/IFIP*. IEEE, 2016, pp. 754–758.

[9] A. Al-Jawad, R. Trestian, P. Shah, and O. Gemikonakli, "Baprobsdn: A probabilistic-based qos routing mechanism for software defined networks," in *Network Softwarization (NetSoft), 2015 1st IEEE Conference on*. IEEE, 2015, pp. 1–5.

[10] R. Yavatkar, D. Pendarakis, and R. Guerin, "IETF RFC 2753: A framework for policy based admission control," 2000.

[11] Mininet openflow virtual network. Accessed on August 1, 2015. [Online]. Available: http://mininet.org

[12] Floodlight openflow controller. Accessed on August 1, 2015. [Online]. Available: http://www.projectfloodlight.org/

[13] CpQD OpenFlow1.3 Software Switch. Accessed on August 1, 2015. [Online]. Available: http://cpqd.github.io/ofsoftswitch13/

[14] S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan, "The internet topology zoo," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 9, pp. 1765–1775, 2011.

[15] Ostinato traffic generator tool. Accessed on June 1, 2017. [Online]. Available: http://mininet.org

[16] C. V. N. Index, "Forecast and Methodology, 2015–2020 White Paper, Cisco, 2016," 2016.

[17] V. Deart, V. Mankov, and A. Pilugin, "HTTP Traffic Measurements on Access Networks, Analysis of Results and Simulation," in *Smart Spaces and Next Generation Wired/Wireless Networking*. Springer, 2009, pp. 180–190.

[18] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[19] T. Zinner, O. Abboud, O. Hohlfeld, T. Hossfeld, and P. Tran-Gia, "Towards QoE Management for Scalable Video Streaming," in *21th ITC Specialist Seminar on Multimedia Applications - Traffic, Performance and QoE*, Miyazaki, Jap, 3 2010.