Rego Mañez, A.; Sendra, S.; García-García, L.; Lloret, J. (2019). Adapting reinforcement learning for multimedia transmission on SDN. Transactions on Emerging Telecommunications Technologies. 30(9):1-15. https://doi.org/10.1002/ett.3643

# Adapting Reinforcement Learning for Multimedia Transmission on SDN

**ALBERT REGO MAÑEZ[1], SANDRA SENDRA[2,1], LAURA GARCIA[1], JAIME LLORET[1]**
[1] Instituto de Investigación para la Gestion Integrada de zonas Costeras, Universitat Politècnica de València. Carrer del Paranimf, 1, 46730 Grau de Gandia, València. Spain.
[2]Dept. of Signal Theory, Telematics and Communications Department (TSTC), Universidad de Granada. C/ Periodista Daniel Saucedo Aranda, s/n. 18071 Granada, Spain

Corresponding author: Albert Rego Mañez (e-mail: alrema91@gmail.com).

**ABSTRACT** Multimedia transmissions require a high quantity of resources to ensure their quality. In the last years, some technologies that provide a better resource management have appeared. Software defined networks (SDNs) are presented as a solution to improve this management. Furthermore, combining SDN with artificial intelligence (AI) techniques, networks are able to provide a higher performance using the same resources. In this paper, a redefinition of reinforcement learning is proposed. This model is focused on multimedia transmission in a SDN environment. Moreover, the architecture needed and the algorithm of the reinforcement learning are described. Using the Openflow protocol, several sample actions are defined in the system. Results show that using the system users perceive an increase in the image quality three times better. Moreover, the loss rate is reduced more than half the value of losses recorded when the algorithm is not applied. Regarding bandwidth, the maximum throughput increases from 987.16 kbps to 24.73 Mbps while the average bandwidth improves from 412.42 kbps to 7.83 Mbps.

**KEYWORDS** Software Defined Network (SDN); Machine Learning; Reinforcement Learning; Artificial Intelligence (AI), Routing; Multimedia; Openflow; Protocol.

## I. INTRODUCTION

Multimedia content has different network resource requirements than other types of content. Ensuring Quality of Service (QoS) and Quality of Experience (QoE) is one of the main concerns of researchers that look for novel ways to guarantee good quality video transmission. Software Defined Networks (SDNs) are presented as a solution to improve the resource allocation management and to provide fairness among users of multimedia streaming adaptive applications [1][2]. SDNs present several advantages over traditional networks that place them as an excellent solution for the challenges of multimedia video streaming. The programing interface in SDN allows a better control over the network, providing an improved performance and more configuration options of the network architecture [3][4]. Furthermore, SDN allows reconfiguring the network flows according to the needs of the users without the actual need of independently configuring all devices. As SDNs are logically centralized, the SDN controller has a global view of the state of the network supporting the dynamic optimization of data flows and the available resources [5][6]. Moreover, QoS can be assigned according to the flow or application in an easy manner selecting a priority according to QoS parameters such as bandwidth, jitter, delay, and packet loss.

Numerous applications such as VoIP, video conferencing, video on demand and gaming require the transmission of multimedia content through the network. These multimedia applications that require a high QoS are increasing throughout the years resulting in diverse challenges. Few or non-existing packet loss must be ensured as well as steady network resources and the delay required by the application [5]. Therefore, the type of multimedia application or content must be considered in order to optimize the resources of the network. Possible changes in the network must be considered as well determining the capacity of the routes as well as the path length so as to decide the best route. Server load balancing is another factor to consider so as to avoid distortion and delay when servers are overloaded. Furthermore, different scenarios may present different challenges to address. Satellite communications must maintain the service of other users while providing QoS to multimedia applications as well as integrating the SDN to the standards of the available satellite networks such as 4G and 5G [7].

Artificial intelligence (AI) is being utilized in countless areas. The centralized control over the network provided by SDNs allows applying AI to analyze and predict the need of network resources and to provide differentiated QoS to users and applications. Reinforcement learning is one of the types of machine learning. It is based on performing different actions

as a function of the state of the system to maximize the output. It is then, a valid approach to provide SDN controllers with the tools to optimize routing tasks and the quality of the video content transmitted through the network. However, reinforcement learning has not been deeply studied as a way to increase QoS in SDNs. And, although it has been proposed by Shih-Chun Lin et al. in [8], there is no a concrete definition of states or actions. Furthermore, there is no explanation on how the controller can implement the proposal and the experiments could be performed as mathematical simulations.

In this paper, a reinforcement learning algorithm for multimedia traffic transmission over SDN is presented. OpenFlow [9] is utilized to implement the SDN controller communication and to perform the actions proposed. Furthermore, the actions that the SDN controller can execute according to the metrics of the network and to ensure QoS are depicted. Several tests have been performed considering different scenarios. The proposal is finally tested by emulating it on Mininet. Mininet has been proved to present similar performance that the one obtained in a real SDN [10].

The rest of the paper is organized as follows. Section 2 presents the related work on SDN-based solutions for multimedia content transmission. The proposal is depicted in Section 3. Section 4 analyzes the proposed reinforcement learning algorithm. The obtained results are discussed in Section 5. Finally, the conclusion and future work is presented in Section 6.

## II. RELATED WORK

Multimedia traffic has been the object of study of numerous researches that looked for increasing QoS and QoE utilizing SDNs for resource allocation. O. Awobuluyi et al. presented in [11] an SDN controller for scalable video encoded over H.265. The information on the scalability of the multimedia streams and network capabilities was known by the controller that made the decisions to ensure QoS and QoE over 5G networks. The proposed architecture was comprised of a Video Quality Assurance Manager (VQAM) that gathered information on the topology and video metrics to decide the routing and video adaptation. It included an SDN Video Quality Orchestrator (SDN-VQO) as well as to manage multimedia flows and to ensure fairness. J. Castillo et al. proposed in [12] a modification of the Entity Title Architecture (ETArch) employing OpenFlow and the QoS control method from SMART (Support of Mobile Sessions with High Transport Network Resource Demand). It provides coordination over dynamic and admission control of super-dimensioned resources in order to provide QoS and QoE to multimedia flows. Tests were performed on a real testbed obtaining the admission of all the sessions with a network saturation up to 170%, reaching a 27% of SSIM improvements over legacy ETArch and reducing the signaling load of legacy ETArch in a 251%. An Network Function Virtualization (NFV) architecture for managing and controlling multimedia applications over 5G with SDN and considering QoE requirements was introduced by A. A. Barakabitze et al. in [13]. NVF and SDN is utilized to reach the Key Performance Requirements/Indicators (KPR/I) of 5G. Different Virtual Machines (VMs) were utilized to address different network requirements of resources of the multimedia applications. The QoE was assessed utilizing the QoE-sdnFlow manager and the QoE-sdnFlow Monitor. The type of multimedia applications was considered to provide the specific resources required by the user. H. Nam et al. presented in [14] an SDN application to monitor the network and assign routing paths to video content utilizing Multi-Protocol Label Switching (MPLS) in order to ensure QoE. The Junos Space SDN platform was utilized to implement the proposal. Tests were performed in different scenarios employing a simulation tool. Results showed an improvement of de viewing experience of 55.9% in peak hours. Lastly, T.-N. Lin et al. presented in [15] a meter-based multicast method to provide end-to-end QoS for multimedia services called OpenE2EQoS. The designed algorithm utilizes a learning mechanism to allocate the available bandwidth for the multimedia flows in order to guarantee QoS. The congestion problem in links is addressed by forwarding low priority packets among N different routes. Therefore, the problem of forwarding the multimedia flows to other routes is prevented. Tests were performed in a real SDN environment were the effectiveness of their proposal is demonstrated.

Optimization has been utilized as well as a tool for increasing QoS and QoE. A. Kassler et al. presented in [16] an SDN-based QoE-driven centralized multi-user path optimization for multimedia services. It maximized QoE considering link capacities, delay, network topology and service utility functions. The user was able to select video quality over audio quality and vice versa. Therefore, video quality may be degraded while audio quality is maintained high enough when the resource availability of the network decreases. OpenFlow was utilized to determine the paths according to the results of the optimization. A new OpenFlow controller called OpenQoS was introduced by H. E. Egilmez et al. in [17]. OpenQoS was intended for multimedia delivery and QoS support. The paths of the multimedia flows are optimized so as to ensure QoS. The performance of the proposed controller was compared to the exiting HTTP-based multimedia adaptive streaming. Results showed that OpenQoS outperformed HTTP-based multimedia adaptive streaming, guaranteeing seamless video delivery in UDP and full video quality in TCP. Á L. Valdivieso et al. proposed in [18] an SDN-based framework to optimize multimedia routing. The framework utilizes NFV and OpenFlow in order to test the routing algorithms. The effectiveness of the framework is determined by the QoS routing algorithm and network performance modules. The parameters analyzed were Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM) and Mean Opinion Score (MOS) from both best-effort and optimized routing. Results showed an improvement in terms of QoS of the proposed framework

compared to the best-effort approach. Finally, E. Grigoriou et al. presented in [19] an SDN-based resource management mechanism for multimedia services that optimizes the QoE of end users. In order to do so, the service was divided into different tasks assigned to the neighbor nodes taking into consideration the final quality. Authors stressed the enhancement of QoE management, resource allocation and video quality that SDN/NFV provided. Tests were performed with the OpenDaylight controller and the Mininet network emulator. Results showed an improvement in video quality.

Although the improvement of QoS through SDNs has been addressed by other researches employing different methodologies, in this paper, reinforcement learning is utilized to address the QoS requirements in multimedia traffic.

## III. A PROPOSED ARCHITECTURE OF MULTIMEDIA TRANSMISSION OVER SDN

This section presents the proposed architecture as well as an overview of the system goals.

### A. *ARQUITECHTURE*

The scenario used in this proposal is a SDN entirely managed by a single controller. This controller must not only decide the path each packet has to follow but also has to take actions to prevent transmission problems. In order to achieve this, the controller is composed by several modules. These modules are shown in Fig.1.

As Fig. 1 shows, the communication module provides the message exchanging between the controller and the network nodes, i.e., the switches. The communication is entirely based on the Openflow standard [9]. Therefore, the network nodes must be Openflow-enabled devices. By using Openflow messages, the controller gathers data about the capabilities of the Openflow-based network nodes. Then, this communication with Openflow messages allows the controller to generate orders with the actions the nodes should perform.

The routing module is in charge of detecting the network topology and creating paths. Moreover, it analyzes the nodes characteristics.

The third module on the SDN controller is the Statistic Analyzer which analyzes the statistics sent by the network nodes to the SDN controller. These statistics are provided by the communication module. The analysis is performed in order to detect the different problems during the multimedia transmission. Although, this is not the main focus of this paper, the module will numerically communicate the transmission problems of QoS degradation. Furthermore, it is the module that notifies changes in the transmission status. These states are defined in the next section.

The last module is the Action Chooser which uses reinforcement learning to determine the action the SDN controller should take in order to solve the current problem that the Statistic Analyzer has detected. It receives the route characteristic from the routing module in order to know which actions can performed in which paths. In addition, the Statistic Analyzer notifies changes in the state of the transmission and its QoS degradation. With this data, the Action Chooser calculates the reward of the last action and choose another action if required. Next section describes the reinforcement learning algorithm and the concepts whose definition is adapted to the problem.

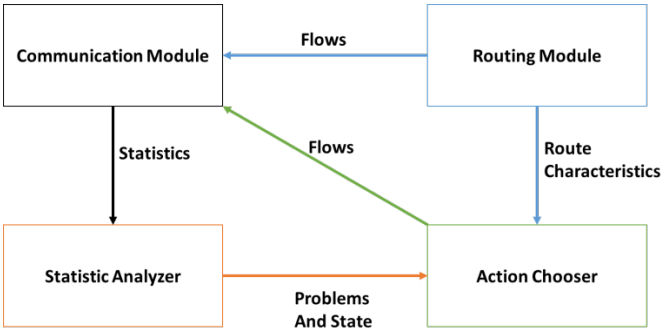


FIGURE 1. SDN Controller Modules

### B. *NETWORK CONSIDERATIONS*

This subsection explains several concepts that the AI module handles. These concepts are needed to correctly evaluate the state of the system.

Fig. 2 shows an example of network we use to test our proposal.

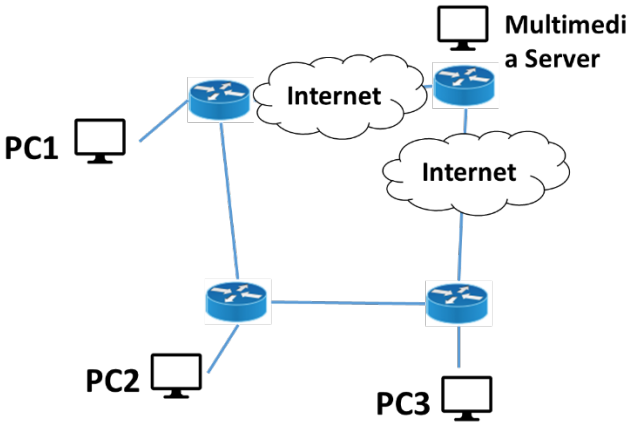


FIGURE 2. Network Example

As we can see, the scenario is composed by a multimedia server and several multimedia clients that will perform the multimedia communication. Although there are proposals that permit multicast flows on SDN [20], in this paper, the multicast transmission will be treated as N unicast flows. This assumption can be done because the system is independent to the unicast or multicast property of the multimedia transmission. In addition, to reach a destination, the flow will go through several nodes from the source to the destination.

This route will be composed by two different links. On the one hand, outside links are the links between the clients and the first network node. On the other hand, inner links are the ones that interconnect two or more network nodes. This differentiation will be useful for the AI module in order to determine which action should be taken. The actions may vary depending on the characteristics of the link. If the link is an inner link, some actions like changing the route would be able to improve the transmission quality. However, if it is an outside link, some other actions like reducing the throughput of the other host could fit better. In addition, some links are backup lines that are not used unless they are required to ensure the QoS requirements of the multimedia transmission.

Nevertheless, the differentiation between links is not the only one done in the system. Depending on the network topology, nodes can also be used as backup nodes or as load balancing nodes. Both of them are nodes that gets to the same intermediate node between the source and the destination. The difference between them is that backup nodes are only used when they are needed. However, the load balancing nodes are nodes that allow the controller to send the packets to an alternative route. The identification of the alternatives routes or backup lines/nodes is the task of the Routing Module.

## IV. REINFORCEMENT LEARNING ALGORITHM

This section describes the reinforcement learning algorithm which is defined in two steps. First, the environment of the system is defined and the states discussed. The system needs some data structures to operate correctly. These data structures are explained then. After that, the concepts about rewards, policy and objective are defined. Later, the actions are described and it is explained how they are performed with the Openflow standard. Finally, the algorithm is depicted.

### *A. Environment and States*

In this subsection, every single element that the algorithm uses is explained.

Firstly, we have to understand that the network is composed by elements with different roles. The most important one is the element that acts as an agent. The agent of our algorithm is the SDN controller. It is the element which will perform the actions and that will analyze the current state of the system. The network nodes are not agents and they will only modify their OpenFlow tables in order to execute the commands of the SDN controller. So, they play a role less important.

The environment of the algorithm is composed by the network and the current state of the network based on the statistics. So, the agent, i.e., the SDN controller, modifies the environment through its actions, although it will actually modify the state of the network. That means that the statistics they gather from the nodes will measure the effects of the performed actions. These actions will change the transmission performance in terms of QoS parameters.

The state of the network is related to the problems suffered by the multimedia transmission. Some of the typical problems are the lack of available bandwidth, a high loss rate, too high delay, too high jitter, etc. The SDN controller receives an alert from the Statistic Analyzer and determines the best action to perform depending on the current state. The current state can be represented as a state-machine (see Fig. 3). The default state for every multimedia flow is the “No Problem” state. Along the transmission of the multimedia stream, several problems may occur and the network will be in charge of adapting its functioning to the new situations. The problems and states defined in this paper are mainly related to QoS parameters. The bandwidth, delay, jitter and loss rate are the parameters the Statistic Analyzer monitors. Besides providing the Action Choose the measures of the problems, it evaluates these parameters on the multimedia flows in order to notify the state changes. When a problem is detected, this leads the system to a new state related to that problem and the system will remain in that state until the problem is solved. Consequently, the system will return to the “No Problem” state. However, some of these problems could be combined and the current state could even change to a worse state. This can happen when the selected action did not generate the required effect or it was performed too late. Fig. 3 shows all these states and changes between them. All those states allow coming back to a better state when the problem is solved (or its QoS parameter is improved).

It is expected that the actions performed change the states permitting the system remains on states with good performance. The best scenario would be remaining in the “No Problem” state. However, in an actual scenario the problems happen and so, it will better to be in states with one problem than being in states with combined problems. In order to achieve this, the following actions can be chosen by the SDN controller:

- Wait
- Switch to a backup line
- Switch to a backup node/path
- Load balancing
- Use QoS-Guaranteed Queues
- Tag packets with VLAN PCP
- Limit the available bandwidth for a flow

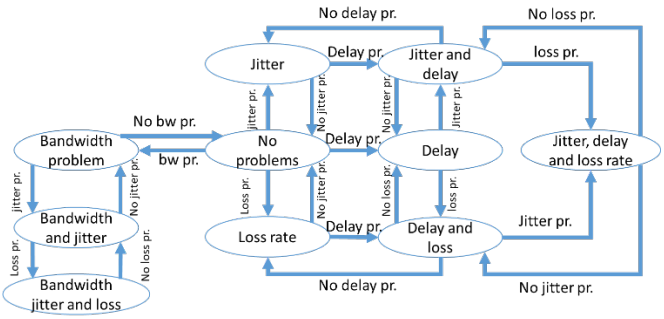

FIGURE 3. Diagram of the states of a multimedia flow in the system.

These actions provoke changes in the network performance, and, depending on the next values obtained from the Statistic Analyzer, the reward value of the action taken will be updated.

The way the rewards are assigned and calculated and how the actions are performed are described in the next subsections. However, we need to firstly know the data structures and concepts.

### B. Data Structure

In this subsection, the data structures that the algorithm needs are described.

The algorithm needs two different data structures in order to work. First, a two-dimensional array it is required to store the rewards of each action-state pair. This array, called H, contains the reward obtained for each combination of action and state. The structure of H is shown in Fig. 4. This structure is independent of the number of multimedia flows. The values shown in the table are examples. Usually, the algorithm should begin with the same value for every entry. However, specific initial values could fit to some scenarios.

| State/action | $a_1$ | $a_2$ | $a_3$ |
|---|---|---|---|
| $s_1$ | 1.3 | 7.5 | 2.21 |
| $s_2$ | 0.4 | 1.2 | 2.05 |
| $s_3$ | 1 | 2 | 0.75 |

FIGURE 4. Example of Array H.

Nevertheless, the system should independently manage each multimedia flow. That means the actions are caused by changes in the state of a single multimedia flow. Keeping the state of each multimedia flow and its route characteristics, it is important to get the highest performance. Therefore, the Multimedia Management array (M) is defined. The array M (shown in Fig. 5) is a two-dimensional array that contains the status of each multimedia flow. Moreover, it is also composed by a set of boolean flags which indicate if the related actions can be performed for that flow or not. Furthermore, the current timestamp is also stored in this array. In Fig. 5, the fields are shown in bits in order to easily see the size needed to store this data All the data related to the capabilities of the route are provided by the routing module.

| State/action | $a_1$ | $a_2$ | $a_3$ |
|---|---|---|---|
| $s_1$ | 1.3 | 7.5 | 2.21 |
| $s_2$ | 0.4 | 1.2 | 2.05 |
| $s_3$ | 1 | 2 | 0.75 |

FIGURE 5. Example of Array M.

### C. Rewards, Policy and Objective Function

The reward obtained due to taking one action or another depends on the problem itself and on the metrics that show how good or bad the performance of our system is. Typically, reinforcement learning is applied to games, especially video games. In those games, the reward obtained depends on the score obtained in that game. Actions like earning a coin or defeating an enemy produce a big reward, because they increase the score.

In our system, the goal is to avoid the multimedia transmission problems. Therefore, the metrics used are the QoS measurements. Consequently, the rewards are assigned depending whether the problems are being reduced or not.

First, it is important to highlight that in order to adapt reinforcement learning to the multimedia transmission in SDN, the rewards must be given attending to the effects that previous actions has produced. That means, that the effects are not suddenly perceived, so the reward from an action must be calculated in several timestamps or iterations. So, the greater effect will not be perceived in the first timestamps. This way of operating, with several intervals, introduces two new definitions, i.e., the time between iterations ($\boldsymbol{\tau_{it}}$) and the number of iterations ($\boldsymbol{n_{it}}$) before calculating the new reward. Then the time ($\boldsymbol{\tau_{rew}}$) the system needs to take an action and to calculate the reward is defined in (1).

$$\boldsymbol{\tau_{rew} = n_{it} * \tau_{it}} \quad (1)$$

Consequently, the reward is an array of ($\boldsymbol{n_{it}}$) elements, one for each timestamp measured. As an example, taking $\boldsymbol{n_{it} = 3}$ the reward will be defined as (2).

$$\boldsymbol{r = [r(1), r(2), r(3)]} \quad (2)$$

Each element $\boldsymbol{r(t)}$ of the array is calculated as a difference in terms of the multimedia transmission problems of the two last timestamps, $\boldsymbol{p(t-1)}$ and $\boldsymbol{p(t)}$ (see Eq. 3).

$$\boldsymbol{r(t) = \rho(t-1) - \rho(t)} \quad (3)$$

, where $\boldsymbol{p(t)}$ is the measurement of multimedia transmission problems at timestamp $\boldsymbol{t}$.

This calculation is done taking into account QoS parameters like delay, jitter and loss rate. In addition, the result is 0 if the difference between both terms is negative. This measurement is calculated and returned by the Statistic Analyzer module.

In order to take the total reward $\boldsymbol{R}$ of an action, the individual rewards for each timestamp $\boldsymbol{r(t)}$ must be multiplied by a weight ($\boldsymbol{t/2^{t-1}}$). Due to the fact that for our system is more important the effect in medium and long term

that the immediate response, the total reward can be expressed by Eq. 4.

$$\boldsymbol{R} = \sum_{t=0}^{n_{it}} \boldsymbol{r(t)} * \frac{\boldsymbol{t}}{\boldsymbol{2^{t-1}}} \quad (4)$$

The weight applied to the individual reward value is identified by γ (see Eq. 5).

$$\boldsymbol{\gamma} = \frac{\boldsymbol{2^{t-1}}}{\boldsymbol{t}} \quad (5)$$

Fig 6 shows the evolution of the value of γ as a function of the timestamps. As we can see, this values start to increase after the second timestamp. That means the action will be more rewarded if there is a medium-term improvement rather than only a short-term improvement. This γ factor has been chosen to manage the first 4 timestamps. If a modification of the system is required or wanted, the γ should be adjusted to the timestamp frame. For example, in order to calculate the reward of actions taken in a long-tern data transmission, (5) should be a linear function.

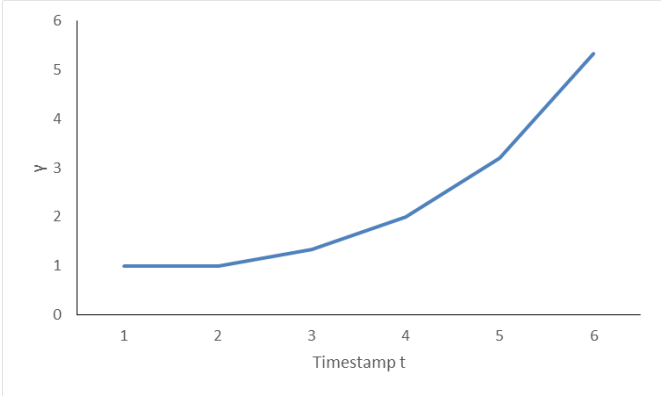


FIGURE 6. γ factor value on each timestamp t.

Regarding the policy the system follows, it consists on choosing the action will offer the maximum reward. The AI module manages a two-dimensional array like the one shown in Fig. 4. When an action has to be taken, the algorithm checks which action, in the current state, has the maximum value, i.e., the reward with the highest value in the row of the current state. This will be the action to carry out. When the reward of this action is obtained, its value is replaced in the array. Thereby, the algorithm is always observing the effects of the actions and learning the best solution to the given problem. The policy function can be described by using Eq. 6.

$$\boldsymbol{\pi(s) = a_i |\, H(s, a_i) \geq H(s, a_j) \forall j \in [1 \ldots M]} \quad (6)$$

, where $\boldsymbol{M}$ is the number of actions defined in the system.

Finally, to understand the goal of the system, we should know the two possible points of view: (1) From the point of view of the states, the goal is to remain the maximum time in a non-transmission-problem state. (2) From a reward-based point of view, the main goal of the system is to obtain the maximum reward possible in each timestamp. Therefore, the policy of the system is to choose the action with highest expected maximum value. If we define the Q-value function under policy π, we obtain (see Eq. 7):

$$\boldsymbol{Q_\pi(s, a) = \sum_{t=0}^{\infty} Max\, R(H, s, t)} \quad (7)$$

Next subsection defines how the different actions can be performed by using the Openflow protocol.

### *D. Actions*

This subsection defines the actions described in subsection IV.A and the message exchange using the Openflow protocol. It is assumed that the actions that involve modification in flows will be only performed on those flows belonging to the multimedia transmission.

All the proposed actions can be executed by using the OFPT_FLOW_MOD message. This message allows modifying or deleting the existing flows in a flow table. Moreover, it allows adding new flows. Its structure is shown in Fig. 7. The more relevant fields for the functioning of our proposal are the command and instructions fields. The command field indicates which action must be done in the table. Adding a new flow, deleting or modifying an existing one are actions performed by the same OFPT_FLOW_MOD message. The instructions are the actions that must be performed when a flow of packets match with the commands. Those instructions enable us to modify the state of the environment. Depending on the required action, the value of the 'command' field and the instructions vary. For each action defined, the instructions needed and the value of the command field is commented.

| OFP_FLOW_MOD | | | | | | |
|---|---|---|---|---|---|---|
| 8 Bytes | 8 Bytes | 1 Byte | 1 Byte | 2 Bytes | 2 Bytes | 2 Bytes |
| cookie | Cookie_mask | Table_id | command | Idle_timeout | Hard_timeout | priority |
| 4 Bytes | 4 Bytes | 4 Bytes | 2 Bytes | 2 Bytes | 8 Bytes | 8 Bytes |
| Buffer_id | Out_port | Out_group | flags | importance | match | instructions |

FIGURE 7. OFP_FLOW_MOD message structure.

The first/second action is to switch to a backup line. The backup line is a dedicated link between nodes that is only used when required. The controller must send an OFPT _FLOW_MOD message to the node that connects with this backup line. In this message the "command" field gets the OFPFC_MODIFY value, and the instruction is OFPAT_OUTPUT. The instruction consists on forwarding packet through the port of the backup line. Therefore, the multimedia transmission will be sent through this backup line without the interfering of other traffic. This, combined with an

adequate network configuration, will ensure an acceptable bandwidth or delay in that section of the network.

The next action is to switch to a backup node. This means that there are one or more network nodes that are going to be bypassed. In order to perform this action, the flow entry related to this flow must be modified. The output port will be changed to the corresponding port that connects with the backup node. In order to perform the action, an OFPT_FLOW_MOD message is sent to the node connected to the backup node. This message is similar to the one sent in the last action but the output port is the corresponding port to get to the backup node. An OFPT_FLOW_MOD with the command set to OFPFC_ADD and the instruction OFPAT_OUTPUT must be send to the backup node. Therefore, the backup node will be able to forward the packets to the next node. The output port corresponds to the one that links with the next node. The next node is the same that performing the action before.

Load balancing is a typical technique that takes advantage of the network redundancy in order to ensure a good level of quality in the multimedia transmission. The load balancing can be performed in different ways. The flow that consumes more bandwidth could be forwarded through the redundant secondary way alone. Another solution would be separate the flows attending to their transport layer protocol. Moreover, the multimedia transmission could be forwarded to both paths, changing the route periodically. In this proposal, the multimedia flow is forwarded to an alternative/backup path. Consequently, an OFPT_FLOW_MOD with the command set to OFPFC_MODIFY and the instruction to OFPAT_OUTPUT is sent to the last node in the original path. The output port is the one that connects to the first node in the alternative path. In addition, an OFPT_FLOW_MOD with the command set to OFPFC_ADD and the instruction to OFPAT_OUTPUT for each node in the backup path is sent. Thereby, the flow is able to follow the path and get the destination.

Another possible action is to use QoS-Guaranteed Queues. These queues are high-priority queues that the network nodes can use to forward packets minimizing the delay and jitter of the multimedia transmission. The action is performed again with the OFPT_FLOW_MOD message, but, in this case, the instruction needed is OFPAT_SET_QUEUE. This Openflow action provides queue selection. The OFPT_FLOW_MOD message is sent to the required nodes where the multimedia problem is located and the queue is set to ensure the QoS required.

The QoS can be handled by using the IEEE 802.1p standard managing the VLAN to differentiate different kind of traffics. Therefore, traffic flows can be managing attending to their requirements. In order to perform this task in our system, the OFPT _FLOW_MOD message uses the OFPAT_PUSH_VLAN instruction. This instruction allows the controller to push a new VLAN tag to the flow. In this tag, the PCP field is indicated. Depending on the value of the PCP field, and the VLAN configuration, the flow can be guaranteed to have a minimum level of QoS.

The last action defined in the system is to limit the available bandwidth of a flow. The Openflow 1.5.1 standard defines meters. Per-flow meters enable OpenFlow to implement rate-limiting. It is a simple QoS operation that constrains a set of flows to a chosen bandwidth. In this case, an OFPAT_METER instruction must be set in the OFPT_FLOW_MOD message. The meter can be configured to constraint either the kbps or the packet/s. In terms of messages, this action can be executed in several ways. On the one hand, the flows that do not contain multimedia content can be modified. An OFPT_FLOW_MOD message will be send with the OFPAT_METER action as an instruction. On the other hand, the OFPT_FLOW_METER instruction can be added as the default instruction. That means that the default flow will contain the meter instruction. The default flow is the one that will match any flow that does not match with the previous entries in the flow table. How this action is implemented is not critical. However, it is important to know the effect of taking this action and it is well-performed.

### *E. Algorithm*

Fig 8 shows the operation algorithm of the Action Chooser which can be divided into four parts, although Fig.8 only shows three of them. Firstly, the algorithm starts initializing the connection with the other modules (Routing, Communication and the Statistic Analyzer) and the data structures (H and M). Then, it listens to changes from both, the Routing Module and the Statistic Analyzer. The secondly, the Action chooser should manage the multimedia flow. To do that, the routing module can notify that a new multimedia flow, called m, has been created. Then, the row for that flow m in the M array is initialized. Attending to the data the routing module has provided, the flags are set. The state and the timestamp are set to zero. If the routing module notifies the end of the multimedia transmission, called m, the data from the M array for that flow is deleted. This is done to maintain the memory efficiency. The third part is the action execution. When the Statistic Analyzer notifies that a multimedia flow m presents a problem in the states, the action should be executed. The state of that flow is updated in the M array and the timestamp and the reward are initialized. The timestamp is set to T, being T the number of iterations for calculating the reward. Then, an action is executed following the policy described in the previous subsection.


FIGURE 8. General AI algorithm.

The fourth task of the algorithm is the reward calculation in each timestamp. This is performed in an independent thread. Fig 9 shows the operation algorithm of the fourth task. This process is performed for each multimedia flow m with timestamp different to zero. For each timestamp the reward is calculated using the measurements obtained from the Statistic Analyzer. The reward is added to the total reward and the timestamp of the flow is discounted by one. When the timestamp reaches zero, the reward in the H array is updated and the reward is set again to zero. Both arrays are critical sections in the code and the concurrency has to be properly managed in order to ensure the system reliability.

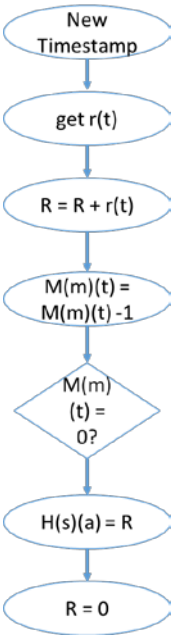


FIGURE 9. Reward calculation algorithm.

### *F. Comparison with other proposals*

To sum up, Table 1 shows a comparison of our proposal with some of the proposals discussed in Section II.

Every proposal uses the statistics gathered from the links to choose the best action or route. Although our proposal does not use the characteristics of the multimedia traffic to perform the actions, the routing module could use them to choose the initial path. Only Kassler [16] uses several characteristics. Awobuluyi [11] uses codec layers and Nam [14] the resolution of the video. The actions their proposals perform are oriented to load balancing, changing the route dynamically. However, our proposal defines a set of actions and it decides, based on previous rewards, which action perform. These actions are not restricted except for the limitations of the southbound interface.

Regarding the scenarios on each proposal can be used, our proposal is the only one that fits any scenario. Furthermore, it is the only one that uses AI techniques to learn. This means, that is the most adaptable proposal and it can be used on every kind of network and conditions. The AI algorithm allows the system to learn the best actions to perform and they may vary according to the scenario. Moreover, the reinforcement learning avoid any supervised learning to be performed.

| | Awobuluyi [11] | Nam [14] | Kassler [16] | Our Proposal |
|---|---|---|---|---|
| **Link stats used** | Yes | Yes | Yes | Yes |
| **Video stats used** | Video Layers | Resolution | Yes | No |
| **Actions taken** | Route changes | Route changes | Route changes | Any Open-flow action |
| **It can be used on…** | Networks with multiple paths | Networks with video servers | Session oriented traffic | Any network |
| **Applied to…** | Multimedia traffic | Video streaming | Any kind of traffic | Any kind of traffic |
| **AI?** | No | No | No | Yes |

TABLE 1. Comparison of the proposals

## V. RESULTS

This section analyzes the model proposed. To do it, several scenarios are proposed and the evolution of the measurements done by the AI module is studied. Moreover, the QoS parameters of the multimedia transmissions are also measured.

### A. *Scenarios and Equipment*

In this subsection, the scenarios and the network tested are presented. Furthermore, the equipment used to perform the tests is detailed.

The network tested is the one presented in Fig 10. The hosts are labeled as PC1 to PC7 and the network nodes are tagged as S1 to S8. This network presents some characteristics that can be used to test the proposal. First, there are multiple paths to reach the destination. For example, a packet sent from PC2 to PC6 can go through S3 or S6. Moreover, there is a backup line between S4 and S8. All the links are set to 100 Mbps. However, the route chosen by the routing protocol to reach PC7 from S1 is through S3. This is because the link between S6 and S7 presents a higher jitter. All the hosts can work as a multimedia client or as a server.

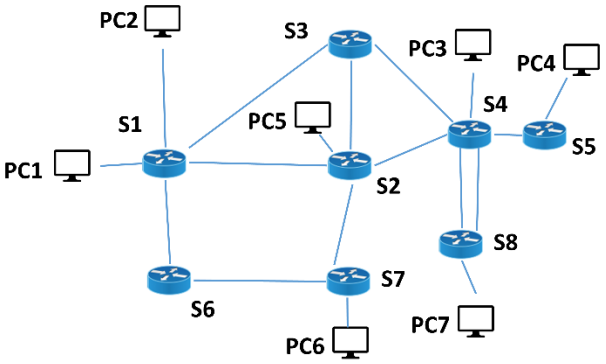


FIGURE 10. Network tested

This network is emulated on Mininet SDN emulator. The characteristics of the equipment used to run the emulations are shown in Fig. 11.

With this environment, the following tests are performed. First, the reward values are measured depending on the value of $\boldsymbol{\tau_{rew}}$ defined in (1). Then, a comparative of the performance of a multimedia transmission between using the algorithm or not is presented. This comparison is done in two different situations. The multimedia transmission of the tests is a video transmission. The video characteristics are also detailed in Fig. 11. For all tests, a QoE evaluation performed by 12 different users is performed. They give a score from 1 to 10 depending on the video quality they perceived to finally calculate the average value. This value is named MOS. The age of the users go from 8 years old to 86 years old. Moreover, their experience with videos is different from one to each other. Two users have little experience with videos, two users work with video and two are studying university degrees related to new technologies. The rest have experience as video consumers.

| Computer Specifications | | | Video Specifications | | |
|---|---|---|---|---|---|
| CPU | Memory | OS | Resolution | Size | Length |
| Intel Core i7-55000U 2.40GHz | 8GB | Ubuntu 18.05 | 1280x720 | 30MB | 120s |

FIGURE 11. Equipment characteristics

### B. *Results of the first test*

In the first scenario, the video is transmitted from PC1 to PC6. In this case, the node S2 is collapsed and that makes hard to ensure the QoS of the multimedia transmission. In this scenario, the time between iterations and the number of iterations to calculate the reward are changed to evaluate its impact. Seven different transmissions are evaluated. The value of $\boldsymbol{\tau_{rew}}$ is changed from 10s to 80s. A value of 70s for $\boldsymbol{\tau_{rew}}$ is not evaluated because the maximum number of iterations is set to 6. The array H is initialized with the values shown in Fig.12 so that the algorithm has to recalculate the rewards in several iterations. Regarding presentation, not all states and actions have been included in the figure, only the relevant ones in this test.

| Actions/State | Backup Line | QoS Queues | Other actions |
|---|---|---|---|
| Bandwidth P | 3 | 6 | 1 |
| Bandwidth and Jitter | 3 | 6 | 1 |
| Bandwidth, Jitter and loss | 3 | 6 | 1 |

FIGURE 12. H array used in the test

The action that could solve the transmission problem is to change to an alternative path because S2 is collapsed, regardless the higher jitter presented in the alternative route. However, the initial values will cause the system to choose a useless action in this case. Thereby, the effect that the timestamp parameters have in the QoS during the learning is studied in the first scenario. The video is transmitted for 120s. Fig. 13 shows the bandwidth evolution obtained in each transmission. The transmissions have been spitted into two figures attending to the value of $\boldsymbol{\tau_{rew}}$. The transmissions with a value of $\boldsymbol{\tau_{rew}}$ from 10 seconds to 40 are presented in Fig. 14 a). In Fig. 14 b), the transmissions with a $\boldsymbol{\tau_{rew}}$ of 50, 60 and 80 seconds are depicted. The problems in the node limit the bandwidth of the transmission. When the algorithm recalculates the reward and chooses the correct action, the bandwidth consumed by the video transmission increases. The results are summarized in Table 1. The minimum bandwidth is 0.97 kbps in all the transmissions. The maximum bandwidth is quite similar in the first five transmissions, over the 22Mbps. However, the last two transmissions have a maximum bandwidth of 11.95 Mbps for 60 s and only 5.70 Mbps for the 80 s of $\boldsymbol{\tau_{rew}}$. The average bandwidth in the first transmissions is also greater than in the last ones. For the 10 s and the 20 s, an average bandwidth of 9.60 Mbps and 13.30 Mbps have been obtained respectively. Nonetheless, for 60s and 80 s, an average of 4.87 Mbps and 1.61 Mbps have been gathered.

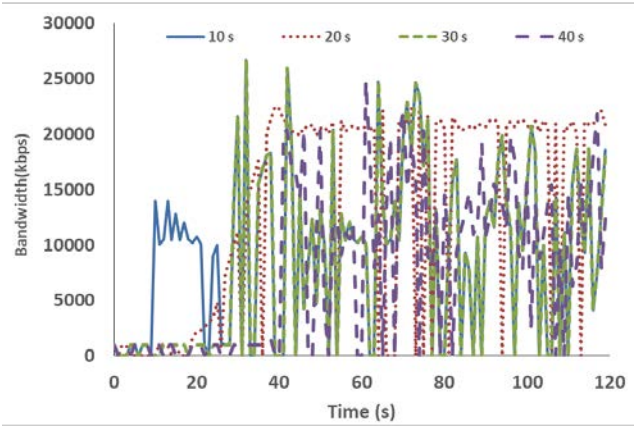


FIGURE 13. a) Bandwidth evolution in the first test ($\tau_{rew}$ from 10 to 40).

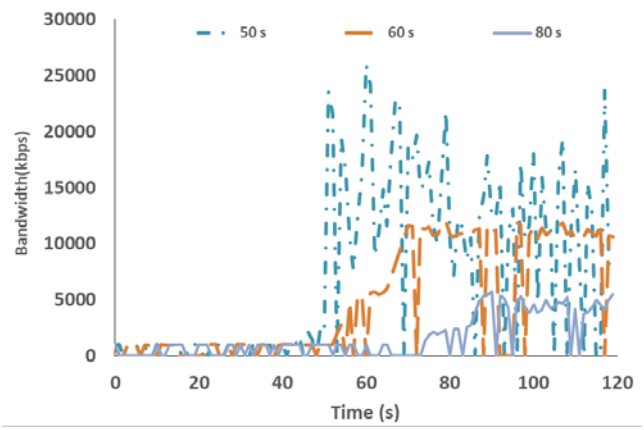


FIGURE 13. b) Bandwidth evolution in the first test ($\tau_{rew}$ from 50 to 80).

| t_rew | Min | Max | Average |
|---|---|---|---|
| 10 s | 0.97 | 26686.08 | 9601.61 |
| 20 s | 0.97 | 22531.30 | 13203.28 |
| 30 s | 0.97 | 26686.08 | 8399.98 |
| 40 s | 0.97 | 24733.44 | 7835.62 |
| 50 s | 0.97 | 25926.72 | 7477.58 |
| 60 s | 0.97 | 11954.50 | 4873.49 |
| 80 s | 0.97 | 5706.04 | 1614.07 |

TABLE 1. Bandwidth statistics for each scenario

Fig. 14 shows the evolution of the delay obtained in each transmission. Fig.14 a) shows the transmissions for a value of $\tau_{rew}$ from 10 to 40 seconds. On the other hand, Fig.14 b) shows the same parameter for the transmissions with a $\tau_{rew}$ value from 50 to 80 seconds. The minimum delay obtained in all the cases is almost 0. The maximum delay for the last transmission is 20.03 ms. For the 30 s transmission, it is 30.08 ms and for the rest, it slightly varies from 22 to 25 ms. The results of delay are summarized in Table 2. Regarding the average delay, it gets higher when $\tau_{rew}$ gets higher. The average delay for the first transmissions is 2.71 ms and 3.09 ms for 10 s and 20 s respectively. However, with 80 s, it is increased to 6.96 ms.

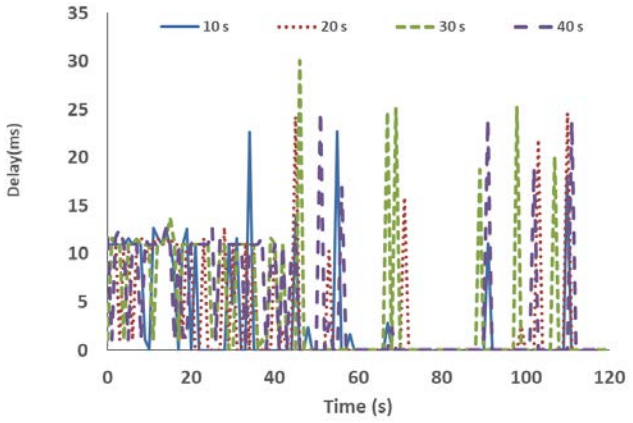


FIGURE 14. a) Delay evolution in the first test ($\tau_{rew}$ from 10 to 40 s).

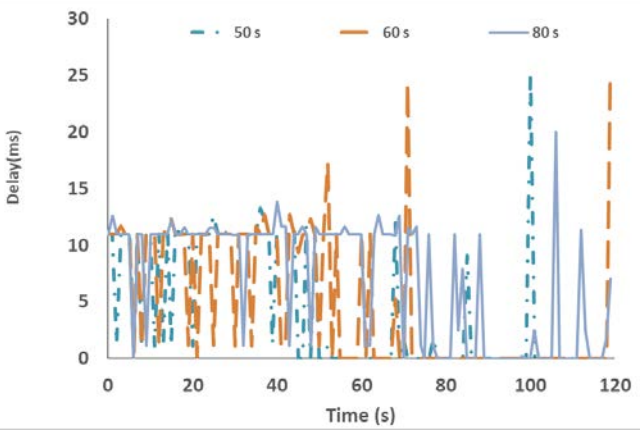


FIGURE 14. b) Delay evolution in the first test ($\tau_{rew}$ from 50 to 80 s).

| τ_rew | Min | Max | Average |
|---|---|---|---|
| 10 s | $3.33 * 10^{-17}$ | 22.71 | 2.71 |
| 20 s | $3.78 * 10^{-17}$ | 24.56 | 3.09 |
| 30 s | $3.35 * 10^{-17}$ | 30.08 | 4.14 |
| 40 s | $3.03 * 10^{-17}$ | 24.81 | 4.26 |
| 50 s | $9.69 * 10^{-19}$ | 25.31 | 4.13 |
| 60 s | $2.79 * 10^{-17}$ | 24.86 | 4.75 |
| 80 s | $3.63 * 10^{-17}$ | 20.03 | 6.96 |

TABLE 2. Delay statistics for each scenario

Fig. 15 shows the values of the jitter obtained during the transmission. Like Fig. 14, Fig.15 a) shows the transmissions for a value of $\boldsymbol{\tau_{rew}}$ from 10 to 40 seconds, while Fig.15 b) shows the same parameter for the transmissions with a $\boldsymbol{\tau_{rew}}$ value from 50 to 80 seconds. The minimum jitter obtained barely varies from the different transmissions. Except for the 20 s and 40 s transmissions, it stays in values close to 0 ms. For these transmissions, the minimum jitter value are 0.55 and 0.14 respectively. The 60 s and 80 s transmission have the lowest minimum jitter values. However, regarding maximum jitter they have the highest values after the 40 s transmission. It has a maximum jitter of 77.89 ms, while 60 s and 80 s transmissions have 69.97 ms and 72.84 ms respectively. The average jitter varies from 10.73 for the 10 s transmission to 14.08 for the 20 s transmission. The transmissions with 60 s and 80 s of $\boldsymbol{\tau_{rew}}$ have an average jitter of 11.43 and 12.94 ms respectively. The results of jitter are summarized in Table 3.

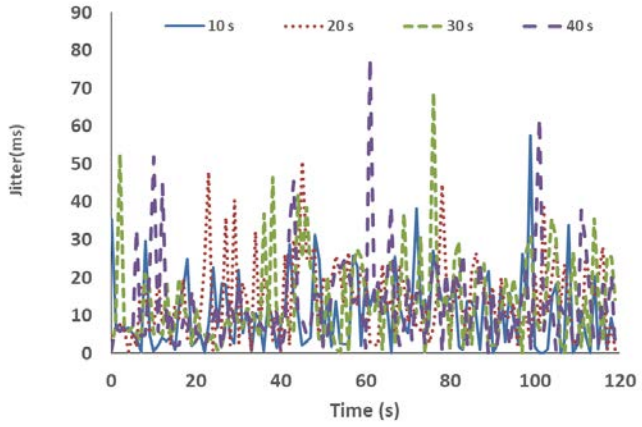


FIGURE 15. a) Jitter evolution in the first test ($\boldsymbol{\tau_{rew}}$ from 10 to 40 s).

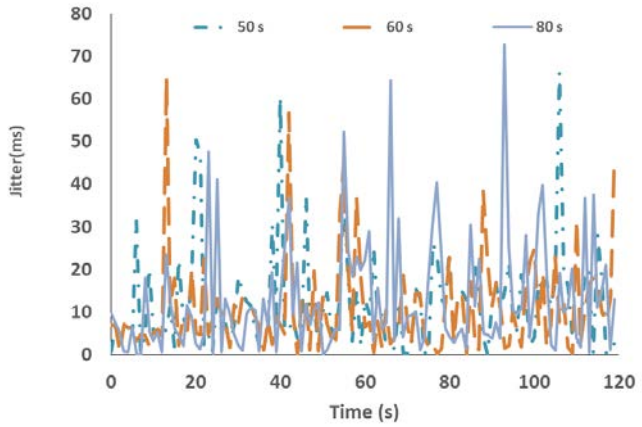


FIGURE 15. b) Jitter evolution in the first test ($\boldsymbol{\tau_{rew}}$ from 50 to 80 s).

| t_rew | Min | Max | Average |
|---|---|---|---|
| 10 s | 0.01 | 57.43 | 10.73 |
| 20 s | 0.55 | 50.69 | 14.08 |
| 30 s | 0.01 | 68.73 | 13.75 |
| 40 s | 0.14 | 77.98 | 13.15 |
| 50 s | 0.02 | 66.20 | 11.96 |
| 60 s | $7.63 * 10^{-16}$ | 69.97 | 11.43 |
| 80 s | $1.00 * 10^{-15}$ | 72.84 | 12.94 |

TABLE 3. Jitter statistics for each scenario

Regarding the loss rate, the results for each transmission are shown in Fig. 16. With 10 seconds, the loss rate is 0.3% and with 20 seconds 0.6%. From 30 seconds to 60 seconds, the loss rate increases from 7.1% to 8.1% respectively. The loss rate of the 40 seconds scenario is 7.9% and 5% from the 50 seconds scenario. Finally, the highest loss rate is obtained when the recalculation takes 80 seconds to be done. This loss rate is 14%. This is caused by the collapse in the node. The node S2 is not able to forward all the data packets and drop many of them. Changing to the backup line earlier reduces the packets drop by S2.

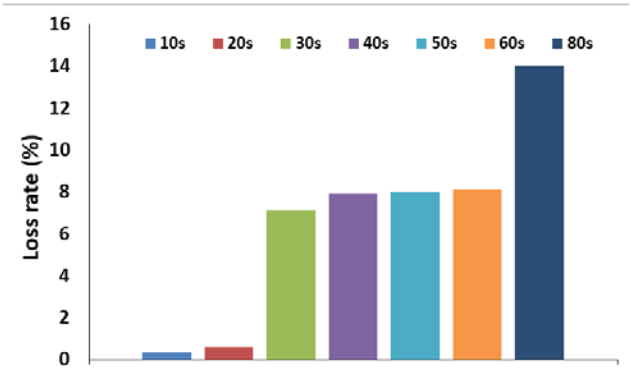


FIGURE 16. Loss rate for each scenario

Finally, the QoE is measured by gathering the MOS of the users. Fig. 17 shows the value of MOS obtained from the opinion of the users. For the transmission with 10 s of $\boldsymbol{\tau_{rew}}$, the score obtained is 7.66 of 10. The 20 s scenario has received a 6.85 and the 30 s a 6. A score of 5.12 and 4.89 has been given to the scenarios with 40 and 50 seconds of $\boldsymbol{\tau_{rew}}$. With 60 seconds of $\boldsymbol{\tau_{rew}}$, a MOS of 4 has been obtained. Finally, the last scenario has received a 2.75 score. The QoE experienced by the users is lower when $\boldsymbol{\tau_{rew}}$ gets higher. This is related to the high loss rate, which causes problems in the transmission such as frozen frames, tiling, noise, ghosting, soft focus, and flickering.

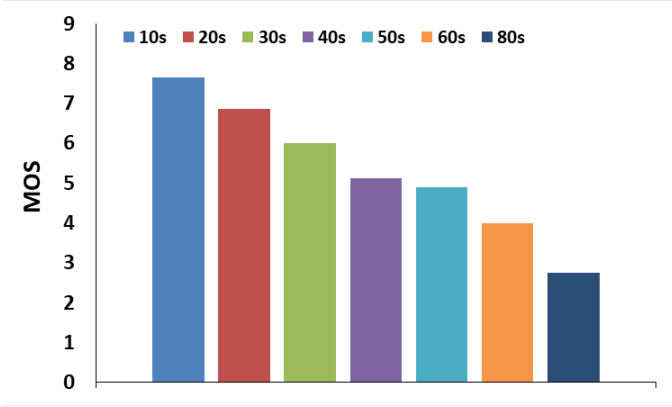


FIGURE 17. MOS obtained in each scenario

### C. *Results of the second test*

In the second test, the video is transmitted again from PC1 to PC6 in similar conditions to the previous scenario. However, in this test, the difference in terms of QoS and QoE between using the proposal and not taking any action is measured. The transmission with the algorithm has a $\tau_{rew}$ of 40 s. This value has been chosen to be not to low nor high. Therefore, it is not an optical parameter value but it provides some improvement to the transmission.

Fig. 18 shows the consumed bandwidth during the video transmission. As we can see, when the algorithm is not used, the minimum bandwidth registered is 0.97 kbps while the maximum value is 987.16 kbps. The average bandwidth during this transmission is 412.42 kbps. On the other hand, when the proposed algorithm is used, the maximum value of bandwidth is 24.73 Mbps while the average bandwidth is around 7.83 Mbps.

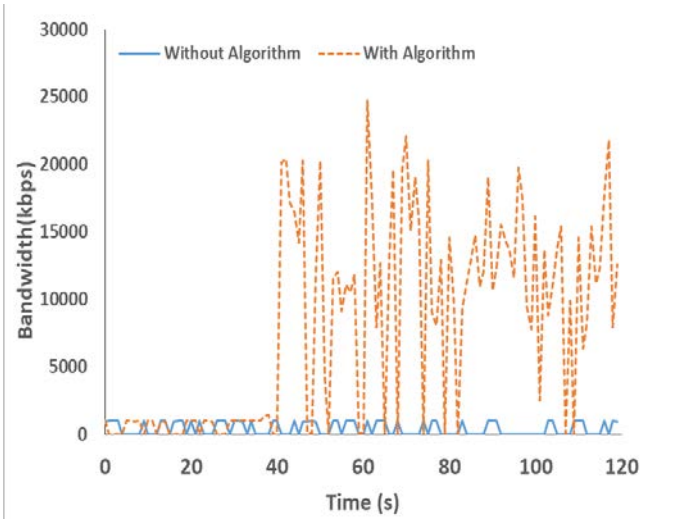


FIGURE 18. Bandwidth evolution in the second test

The delay registered during the video transmission is shown in Fig. 19. The transmission without the algorithm presents a maximum delay of 13.32 ms and an average of 8.94 ms. However, using the algorithm, the maximum is 24.81 ms and the average is 4.26 ms. Without the algorithm, the delay relays more stable, although higher. Nevertheless, using the algorithm, the transmission suffers some delay peaks. This is translated into some little video errors like ghosting and black pixels during the transmission.

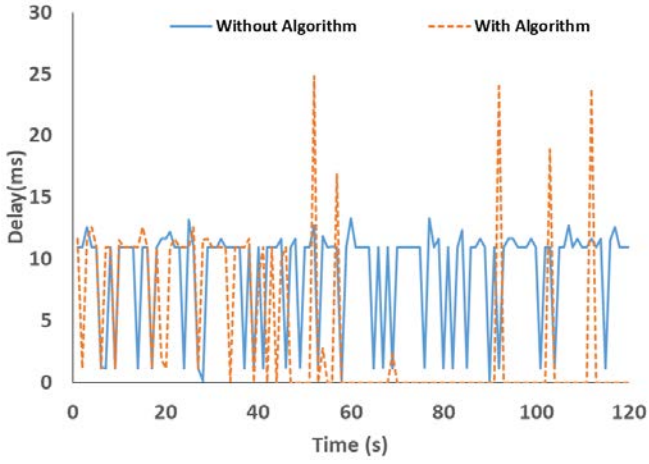


FIGURE 19. Delay evolution in the second test

Regarding the jitter, Fig. 20 shows the difference between the two transmissions for the both cases, i.e., when the algorithm is used and when it is not used. Without algorithm, the maximum jitter is 78.76 ms and the average is 12.34 ms. With the algorithm, the maximum jitter is 77.98 ms and the average 13.15 ms. Both transmission have a stable jitter, with some peaks Therefore, both jitter graphs are quite similar, with only an increment of 6% of the average jitter. Furthermore, the jitter peaks produced when the algorithm is used are due to the lower performance of the alternative route.

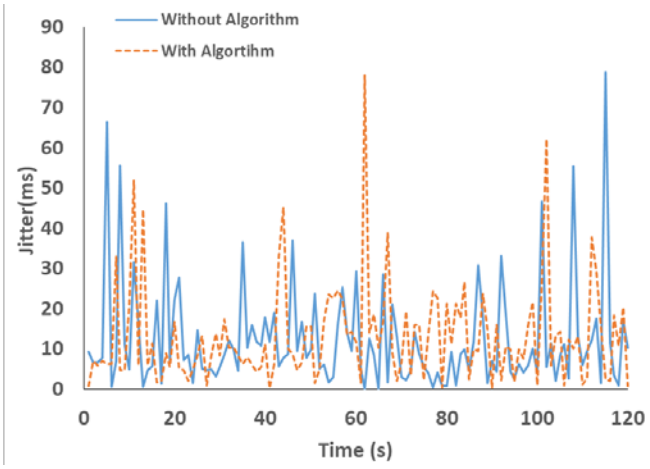


FIGURE 20. Jitter evolution in the second test

The loss rate is also compared for both cases. Fig. 21 shows this comparison. In blue, the transmission when the algorithm is not used presents a 20.5% of loss rate. Nevertheless, when using the algorithm, the loss rate is reduced up to 7.9%.

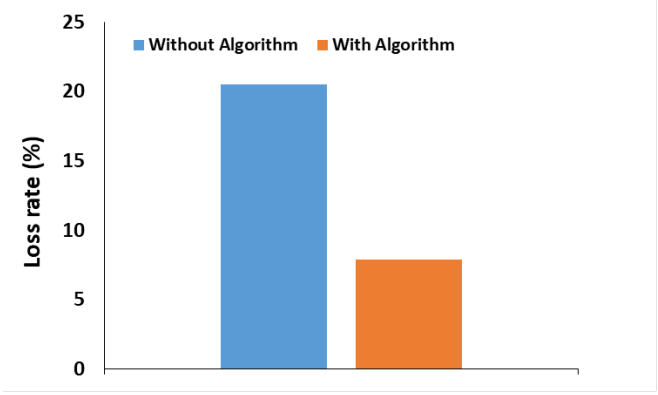


FIGURE 21. Loss rate obtained in the second test

Regarding to the QoE results (see Fig. 22), the average value of all the 12 requested users shows that when the algorithm is not used, the MOS has a value of 1.66 points over 10 while using the algorithm improves in great measurement the results with a value of 5.12 over 10 points. This shows the increment of performance that the action taken by the algorithm provides. Despite the lower performance of the alternative route, the quality is increased when the problem is handled by the algorithm.

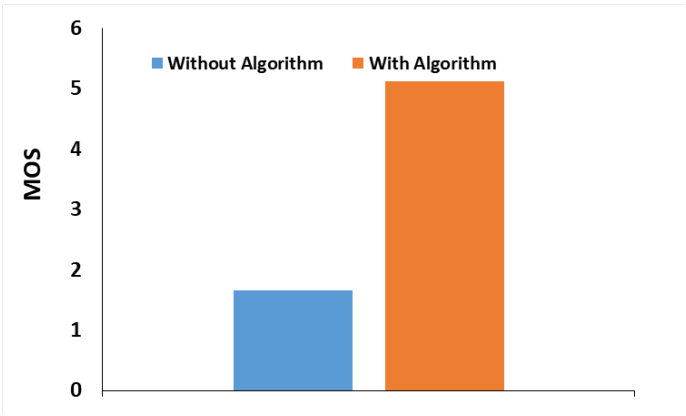


FIGURE 22. MOS gathered for the second test

## VI. CONCLUSION AND FUTURE WORK

Multimedia transmissions require an important availability of resources to ensure an acceptable quality. So, the way the networks manage their resources have a critical impact on this quality. SDN allows managing the resources in a more efficient way. Using AI, networks can be aware of the problems and learn how to solve them in order to provide the best QoS and QoE in multimedia transmissions. Machine learning is a technique that fits the network resource management problem. So, taking into account the aforementioned issues, in this paper, we have presented an adaptation of this technique to SDN. The results obtained show that the introduction of the proposed algorithm improves the quality of the multimedia transmission. In terms of QoE, users perceive an increase in the image quality three times better while the loss rate is reduced more than half the value of losses recorded when the algorithm is not applied. Regarding bandwidth, the maximum throughput increases from 987.16 kbps to 24.73 Mbps while the average bandwidth improves from 412.42 kbps to 7.83 Mbps.

Although these results show interesting improvements in multimedia transmission, some considerations should be taken into account. Firstly, the proposed system is a preliminary solution to these problems. The parameters defined are used in the system but their values may change to adapt the system to the problem. Depending on the problem or the evolution of the Openflow protocol, the states and actions the system manages may vary. Furthermore, the parameters that control the total amount of time needed to recalculate the rewards can have different values. This affects the quality of the multimedia transmission as the results of the first scenario show. The MOS of the transmission with quicker reactions to the problem are higher, i.e., from 7.66 with the quickest reaction to 2.75 with the slowest one. Furthermore, the average bandwidth of the transmission is high. With 10 or 20 seconds of $\boldsymbol{\tau_{rew}}$, the average bandwidth consumed is between 9 and 13 Mbps. However, increasing $\boldsymbol{\tau_{rew}}$, the system reduces the bandwidth

up to 1.6 Mbps. Loss rate also increases from 0.3% and 0.6% to 14%.

As future work, some reconsideration about the proposal can be done. Proposing some default values for the H array for different environments can help to avoid or reduce unnecessary learning. This could improve the QoS and QoE of the first multimedia transmission that face transmission problems. The policy function can be further detailed using old rewards calculations in order to change the rewards values within a range each time. In order to learn and explore all the actions, an extra parameter could be defined and studied. Moreover, how the system can be adapted to networks with different data transmission is an interesting problem that can be studied in the future. Our future work with this proposal will be the total integration with the routing module and its application to other types of data form environmental monitoring in WSN (Wireless Sensor Networks) [21], especially, the ones where video transmission is required [22] and IoT (Internet of Things) [23]. This may imply the definition of new actions, states and data structures.

## ACKNOWLEDGMENT

This work has been partially supported by the “Ministerio de Educación, Cultura y Deporte”, through the “Ayudas para contratos predoctorales de Formación del Profesorado Universitario FPU (Convocatoria 2015)”. Grant number FPU15/06837, by the "Ministerio de Economía y Competitividad" in the "Programa Estatal de Fomento de la Investigación Científica y Técnica de Excelencia, Subprograma Estatal de Generación de Conocimiento" within the project under Grant TIN2017-84802-C2-1-P and by the“Ministerio de Economía y Competitividad”, through the “Convocatoria 2016 - Proyectos I+D+I - Programa Estatal De Investigación, Desarrollo e Innovación Orientada a los retos de la sociedad” (Project TEC2016-76795-C6-4-R). This work has also been partially supported by European Union through the ERANETMED (Euromediterranean Cooperation through ERANET joint activities and beyond) project ERANETMED3-227 SMARTWATIR.

## REFERENCES

1. M. Taha, L. García, J.M. Jimenez and J. Lloret. SDN-based throughput allocation in wireless networks for heterogeneous adaptive video streaming applications. 13th International Wireless Communications and Mobile Computing Conference, Valencia, Spain, 26-30 June 2017; 963-968. Doi: 10.1109/IWCMC.2017.7986416.
2. Jose Miguel Jimenez, Oscar Romero, Albert Rego, Avinash Dilendra, Jaime Lloret. Study of Multimedia Delivery over Software Defined Networks, Network Protocols and Algoritms. 2015. 7: 37-62.
3. Vandana and H. D. Kallinatha. An Enhanced method for Streaming Multimedia data to achieve the Quality of Service Support using SDN. 2nd International Conference on Emerging Computation and Information Technologies, Tumakuru, India, 15-16 December 2017; 1-7. Doi: 10.1109/ICECIT.2017.8453366.
4. A. Rego, S. Sendra, J. M. Jimenez and J. Lloret. OSPF routing protocol performance in Software Defined Networks. 2017 Fourth International Conference on Software Defined Systems (SDS), Valencia, 2017; 131-136. Doi: 10.1109/SDS.2017.7939153
5. M. Karakus and A. Durresi. Quality of Service (QoS) in Software Defined Networking (SDN): A survey. Journal of Network and Computer Applications. 2017; 80:200-218. Doi: 10.1016/j.jnca.2016.12.019.
6. A. Rego, L. Garcia, S. Sendra, and J. Lloret. Software Defined Network-based control system for an efficient traffic management for emergency situations in smart cities. Future Generation Computer Systems 2017; 88:243-253. Doi: https://doi.org/10.1016/j.future.2018.05.054
7. R. Ferrús, H. Koumaras, O. Sallent, G. Agapiou, T. Rasheed, M. A. Kourtis, C. Boustie, P. Gélard and T. Ahmed. SDN/NFV-enable satellite communications networks: Opportunities, scenarios and challenges. Physical Communication. 2016; 18:95-112. Doi: 10.1016/j.phycom.2015.10.007.
8. S. Lin, I. F. Akyildiz, P. Wang and M. Luo. QoS-aware Adaptive Routing in Multi-layer Hierarchical Software Defined Networks: A Reinforcement Learning Approach. IEEE International Conference on Services Computing, San Francisco, CA, USA, 27 June -2 July 2016; 25-33. Doi: 10.1109/SCC.2016.12.
9 Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, Jonathan Turner- OpenFlow: enabling innovation in campus networks. ACM SIGCOMM Computer Communication Review. 2008; 38: 69-74. Doi: 10.1145/1355734.1355746
10. Jimenez, JM.; Romero Martínez, JO.; Rego Mañez, A.; Lloret, J. Analyzing the Performance of Software Defined Networks vs Real Networks. International Journal On Advances in Networks and Services 2016; 9(3-4):107-116. http://hdl.handle.net/10251/83652
11. O. Awobuluyi, J. Nightingale, Q. Wang and J. M. Alcaraz-Calero. Video Quality in 5G Networks: Context-Aware QoE Management in the SDN Control Plane. IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing, Liverpool, UK, 26-28 October 2015; 1657-1662. doi: 10.1109/CIT/IUCC/DASC/PICOM.2015.250.
12. J. Castillo, A. Neto, F. Silva, P. Fonsi, R. Aguiar, A. Ishimori, F. Farias and A. Abelém. Additions to the ETArch Control Plane to Support Multimedia QoS-Guaranteed Content Transport over OpenFlow-Enabled SDN Future Internet Systems. 6th IEEE International

Workshop on Management of Emerging Networks and Services, Austin, TX, USA, 8-12 December 2014; 172-177. Doi: 10.1109/GLOCOMW.2014.7063426.

13. A. A. Barakabitze, L. Sun, I. Mkwawa and E. Ifeachor. A Novel QoE-Aware SDN-enabled, NFV-based Management Architecture for Future Multimedia Applications on 5G Systems. Proceedings of the 8th International Conference on Quality of Multimedia Experience, Lisbon, Portugal, 6-8 June 2016.
14. H. Nam, K. Kim, J. Y. Kim and H. Schulzrinne. Towards QoE-aware Video Streaming using SDN. Communications Software, Services and Multimedia Symposium, Austin, TX, USA, 8-12 December 2014; 1317-1322 Doi: 10.1109/GLOCOM.2014.7036990.
15. T. Lin, Y. Hsu, S. Kao and P. Chi. OpenE2EQoS: Meter-based Method for End-to-End QoS of Multimedia Services over SDN. IEEE 27th International Symposium on Personal, Indoor and Mobile Radio Communications, Valencia, Spain, 4-8 September 2016; 1-6. Doi: 10.1109/PIMRC.2016.7794972.
16. A. Kassler, L. Skorin-Kapov, O. Dobrijevic, M. Matijasevic and P. Dely. Towards QoE-driven Multimedia Service Negotiation and Path Optimization with Software Defined Networking. 20th International Conference on Software, Telecommunications and Computer Networks, Split, Croatia, 11-13 September 2012; 1-5.
17. H. E. Egilmez, S. T. Sane, K. T. Bagci and M. Tekalp. OpenQoS: An OpenFlow Controller Design for Multimedia Delivery with End-to-End Quality of Service over Software-Defined Networks. Proceedings of the 2012 Asia Pacific Signal and Information Processing Association Annual Summit and Conference, Hollywood, CA, USA, 3-6 December 2012; 1-8.
18. A. L. Valdivieso Caraguay, J. A. Puente Fernández and L. J. García Villalba. Framework for optimized multimedia routing over software defined networks. Computer Networks. 2015; 9:369-379. Doi: 10.1016/j.comnet.2015.09.013.
19. E. Grigoriou, A. A. Barakabitze, L. Atzori, L. Sun and V. Pilloni, An SDN-approach for QoE management of multimedia services using resource allocation. IEEE ICC 2017 Communications Software, Services, and Multimedia Applications Symposium, Paris, France, 21-25 May 2017; 1-7. Doi: 10.1109/ICC.2017.7997261.
20. S. Islam, N. Muslim and J. W. Atwood. A Survey on Multicasting in Software-Defined Networking. IEEE Communications Surveys & Tutorials 2018; 20:355-387. doi: 10.1109/COMST.2017.2776213

21 I. Arfaoui, N, Boudriga and K. Trimeche. A WSN Deployment Scheme under Irregular Conditions for Surveillance, Applications. Adhoc and sensors wireless networks. 2017; 35:217-259.

22 I. Mateos-Cañas, S. Sendra, J. Lloret, J. M. Jimenez. Autonomous video compression system for environmental monitoring. Network Protocols and Algoritms. 2017; 9:48-70.

23 S Bakhshad, R. Md Noor, A. Akhunzada, T. Saba, I. Bin Ahmedy, F. Haroon and B. Nazir. A Dynamic Replication Aware Load Balanced Scheduling for Data Grids in Distributed Environments of Internet of Things. Adhoc and sensors wireless networks 2018; 40: 275-296.