Multimedia Systems (2018) 24:313–325
https://doi.org/10.1007/s00530-017-0553-x

REGULAR PAPER

OpenIPTV: a comprehensive SDN-based IPTV service framework

Reza Mohammadi[1] · Reza Javidan[1] · Manijeh Keshtgari[1]

Received: 16 September 2016 / Accepted: 12 May 2017 / Published online: 24 May 2017

Abstract IPTV is an emerging TV content delivery service that should guarantee Quality of Service (QoS) to deliver television contents over IP for their customers. However, providing such QoS regarding service level agreements (SLA) requires frequent service monitoring and adaptive configuration mechanisms. Nowadays, Software Defined Networks (SDNs) provide capabilities to deploy and manage networks dynamically and can maintain QoS. In this paper, a novel IPTV service framework (OpenIPTV) is proposed, which utilizes SDN as an underlying technology for providing QoS for IPTV customers in a shared backbone network. OpenIPTV is implemented in a well-known OpenDayLight controller and strictly followed a modular design for the sake of efficiency. OpenIPTV comprises all service requirements such as resource monitoring, channel changing, multicast group managing and dynamic QoS multicast traffic engineering. The performance of OpenIPTV is evaluated under different scenarios and experimental results confirmed the effectiveness of the proposed framework in terms of QoS metrics. Furthermore, experimental results show that OpenIPTV is a feasible and practical solution to deliver IPTV services with high level of QoS over SDN.

Communicated by T. Plagemann.

✉ Manijeh Keshtgari
keshtgari@sutech.ac.ir

Reza Mohammadi
r.mohammadi@sutech.ac.ir

Reza Javidan
javidan@sutech.ac.ir

1 Department of Computer Engineering and Information Technology, Shiraz University of Technology, Shiraz, Iran

Keywords IPTV · SDN · Quality of service · IPTV over SDN · OpenIPTV

1 Introduction

Traditionally, delivery of TV content has been achieved using satellite-based infrastructure, which causes satellite capacity exhaustion and increases competitive pressures and operating costs to lease a TV channel from network providers [1]. In recent years, multimedia traffic such as IPTV is rapidly growing and it is predicted that 73% of all IP traffic will be video by 2017 [2]. According to [3], the number of global IPTV subscribers has been grown from 26.7 million in 2009 to 81 million in 2013. Furthermore, the Global IPTV market has been grown from $6.7 billion in 2009 to $19.9 billion in 2013 [3]. This causes IPTV to be as an emerging multimedia service to deliver digital television services using IP over a packet-switch network [4].

Generally, IPTV services can be classified into three main categories: Video on Demand (VoD), time-shifted and live television [5]. In VoD, users can request and watch the target video at the desired time from video server, rather than watching at a specific time [6]. In time-shifted service, time shifting of video content is possible and users can replay previous videos or replay current video content from its beginning. In this service, the subscribers watch TV content after some delay [7]. The Live television service is a real-time service and delivers video content to IPTV subscribers and allows them to watch the video content by joining to the multicast group of the TV channel. IPTV network providers can either allocate a dedicated network to carry IPTV traffic, or utilize a common network to carry all traffics including IPTV and non-IPTV traffics [3]. Though allocating a dedicated network can guarantee

IPTV services regarding SLA, it needs more costs, which may not be economical.

IPTV is a paid subscription and it is essential for IPTV providers to deliver IPTV services to their subscribers regarding to SLA. Therefore, a key challenge in IPTV network is to guarantee QoS for subscribers based on SLA. Current IP-based solution regarding IPTV services relies on Internet Group Management Protocol (IGMP) [8] for controlling multicast group; and well-known multicast protocols such as MOSPF [9], PIM [10], DVMRP [11] and CBT [12] for multicast routing between IPTV server and subscribers. Current IP-based architecture for IPTV suffers from some limitations for providing QoS. Because IGMP controls the multicast group, it needs to be configured on the routers over the network. Then, each router should communicate with the other routers to notify events related to multicast group [13]. In an IPTV network, the subscribers may constantly alternate between channels or switch from one channel to another channel, i.e., leaving from a channel and joining to another channel. These behaviors of users change the multicast groups; hence, the multicast routing protocols should exchange many signaling message to reach a consistent state related to the multicast tree. Moreover, IP multicast protocols depend on the supports of network infrastructure and have some limitations on reliability, scalability, security and maintenance signaling [1, 14].

Software Defined Networks (SDNs) provide capabilities to deploy and manage networks dynamically. SDN separates data plane from control plane and facilitates network monitoring and management [15]. In SDN, a logically centralized controller has a global view of network and collects information of network resources to make configuration and routing decisions which push back to the data plane (switches) as a set of forwarding rules [16]. Afterward, the forwarding rules are installed on the flow table of data plane switches that forward traffics based on the forwarding rules [17]. The separation of control and data plane together with programmable nature of SDN, allows using it as an underlying technology for delivering IPTV services in an efficient manner. OpenFlow (OF) is one of the most common open standards for SDN architecture, which contains many features for network monitoring and reconfiguration and allows network providers as well as researchers, to add functionality for networking experiments [18].

Providing high level of QoS is crucial for maintaining subscriber relationships, hence real-time monitoring and traffic engineering in IPTV is a key concern. This paper proposes OpenIPTV as a new framework for delivering IPTV services based on SDN architecture. OpenIPTV takes the advantages of OF and manages multicast groups and finds efficient multicast trees between IPTV server and each multicast groups regarding QoS metrics. OpenIPTV is modular and can manage multicast groups and handles traffic engineering for each group. Managing groups involves join/leave/query messages for establishing multicast group memberships. Moreover, traffic engineering modules first gather links information include link delay and utilization, then models the multicast routing as a Delay-Constrained Least-Cost (DCLC) problem [19] and solves it to compute efficient multicast tree for each multicast group. In the end, this module sends the multicast tree as a set of forwarding rules to the relevant switches in the network. Generally, packet loss and end-to-end delay are two important QoS parameters in multimedia applications such as IPTV that might adversely affect the quality of received video at end users [20–22]. Hence, modeling the multicast routing as a DCLC problem guarantees end-to-end delay QoS metric for IPTV subscribers and reduces packet loss ratio of IPTV contents. Most of the current solutions for delivering IPTV services over SDN require changing the structure of switches or messages. One of the main advantages of OpenIPTV is its ability in implementation over current OF-based networks without changing the OF equipment or packet type. The performance comparison of OpenIPTV with other well-known solutions such as Multiflow [13] shows that OpenIPTV performs better in terms of QoS metrics.

The main contributions of this paper can be summarized as follows:

- An SDN-based framework for delivering IPTV service called OpenIPTV is proposed.
- The proposed OpenIPTV is implemented in well-known OpenDayLight controller; without altering existing technology.
- The promising QoS metrics in IPTV service are modeled as a DCLC problem.
- The performance of OpenIPTV is evaluated and compared with Multiflow in different scenarios in terms of important IPTV QoS metrics to show the effectiveness of the proposed framework.

The remainder of this paper is organized as follows. In Sect. 2, some related works of IPTV over SDN are reviewed. The architecture of proposed OpenIPTV framework together with controller design and traffic engineering issues are described in Sect. 3. In Sect. 4, simulation details and experimental results are presented. Finally, conclusion is described in Sect. 5.

2 Related works

Literatures about multicasting techniques and IPTV service over SDN is reviewed in this section. McDonagh and et al. [23] proposed an architecture for IPTV service based on OF and investigated how OF can assure service quality. In the proposed architecture, core network consists of multiple core routers and edge routers. IPTV server and subscribers have access to core network via edge routers. Each IPTV subscriber has a Set-Top-Box which is connected to a Digital Subscriber Line Access Multiplexers (DSLAM). There is a Control Management (CM) device which acts as a controller and manages the network based on the collected statistics from all devices include core and edge routers, DSLAM and Set-Top-Boxes. The proposed architecture has many unresolved problems such as communication between the DSLAMs and the CM or communication between the Set-Top-Boxs and the CM. Moreover, the authors have not simulated the whole details of the proposed architecture.

Thorpe et al. [17] developed an OpenFlow IPTV network and evaluated the performance of the network in terms of QoS metrics. The authors proposed some new OF messages to add existing OF protocol for delivering IPTV services. Furthermore, they proposed a new OF node model instead of current OF switches to manage and deliver IPTV services over SDN. In their method, the controller computes the best route using Dijkstra algorithm and IPTV server streams the video to destinations through the shortest routes. A key limitation of their proposed method is that it requires to change current architecture of OpenFlow.

Bondan et al. proposed a clean-slate approach for multimedia multicasting in OpenFlow networks called Multiflow [13]. The aim of Multiflow is to manage multicast group efficiently and reduce the time between group joining and receiving video data. In Multiflow, video streaming server sends an IGMP query to the network controller and then controller stores it in a list called *active groups list*. Each interested client to join a group sends an IGMP join message to the controller. Afterward, the controller computes multicast tree using Dijkstra algorithm and inserts the necessary forwarding rules to the relevant switches to establish routes between the video server and receivers. The authors extend Multiflow for a very simple IPTV scenario; but they have not evaluated and analyzed the performance of Multiflow for IPTV services.

Marcondes et al. introduced CastFlow [24] as a multicast framework based on OF. As the authors of CastFlow mentioned, multicast group management of CastFlow does not exist in real situation. Therefore, the authors implemented a separate module for simulating multicast group management. CastFlow computes the Minimum Spanning Tree (MST) using the Prim algorithm and the weight of each edge is defined as the path cost distance between the source of the multicast group and the edge. Although the authors claimed that the CatFlow can be extended for IPTV service; they have not implemented and evaluated CastFlow for a real IPTV scenario. Furthermore, CastFlow needs to be redesigned and changed to qualify as an IPTV service framework.

Rattanawadee et al. presented an IPTV multicasting application based on Multiflow and compared the transmission time of the first joint/receive packet to an IPTV subscriber when using Dijkstra and Prim algorithms [25]. Their simulation results for IPTV show that the transmission time for switching from a channel to another channel is reduced by using Dijkstra algorithm rather than Prim's algorithm.

Noghani and Sunay [2] introduced a framework for multimedia multicasting over SDN. Their framework calculates minimum cost multicast tree using MiniMax algorithm. Their experimental results show that MiniMax incurs low packet loss. A fault-tolerant IP multicasting technique over SDN is presented in [26]. The authors developed a method which computes two different multicast trees between source and destinations. Once a switch in primary multicast tree fails, the controller changes the forwarding rules of network to the alternative multicast tree. In [27], the authors presented a network-layer single-source multicast framework called Lcast. Lcast is an inter-domain multicast framework which uses Locator/ID Separation Protocol (LISP) overlay router to make the network scalable.

None of the discussed and related works, implemented and presented the whole details of IPTV service delivering over SDN; they just proposed some partial solutions for delivering IPTV service over SDN and did not evaluate the solutions under different scenarios in terms of important QoS metrics such as PSNR, Pre-roll, delay and packet loss. As a result, in this paper, a comprehensive framework for IPTV service delivering over SDN is proposed, to show the superiority of the proposed framework; it is evaluated in different scenarios in terms of QoS metrics.

3 OpenIPTV architecture and design

The proposed OpenIPTV framework is described in this section. In the first step, its architecture is introduced, then QoS multicast routing protocol is discussed. Finally, the design of OpenIPTV controller and the functionality of its modules are explained.

3.1 OpenIPTV architecture

OpenIPTV is an SDN-based IPTV service framework, which attempts to deliver IPTV services based on QoS metrics in an efficient manner. To achieve this goal, OpenIPTV should effectively manage multicast groups and perform QoS multicast routing for each multicast group. As mentioned in Sect. 1, thanks to the OF capabilities, the

IPTV network providers can monitor and manage their networks regarding to QoS metrics. Figure 1 shows the architecture of OpenIPTV for delivering IPTV services over SDN. As depicted in this figure, OpenIPTV includes multiple components and devices. IPTV server is originating point which streams TV content over IP. TV contents can be arrived from local video storages, live cameras, via terrestrial or satellite links. IPTV server encapsulates TV content in IP packets and sends the packets to backbone network. The backbone network is not limited to only IPTV services. This means that the other non-IPTV users can also use it for sending their own traffics. Therefore, different types of traffics from different services can be transmitted in the backbone network. Moreover, home subscribers can receive and watch the IPTV contents using STB devices. STB is a device that decodes and decrypts TV contents for TV screen. IPTV subscribers select their desired channels and then IPTV flows are streamed to the STBs through the DSLAM and home routers. The DSLAM is shared between multiple users and receives IPTV traffic from edge OF switch and propagates the received traffic to the STBs of multiple users. Home router can route the TV content between multiple subscribers in a building. Furthermore, in this architecture, other IPTV subscribers can connect to IPTV server from their own network and use IPTV services using IP protocols. The backbone network includes multiple OF switches, which are controlled by an OpenIPTV controller. In OpenIPTV, controller is responsible for multicast group management and QoS multicast routing calculations.

3.2 QoS multicast routing protocol

Developing an efficient algorithm for calculating efficient multicast tree for each multicast group is essential in IPTV service delivery. As discussed in Sect. 2, most current solutions utilize Dijkstra algorithm for calculating Shortest Path Tree (SPT). Although the SPT has minimum cost routes to each destination, it does not guarantee the end-to-end delay between source and destinations in multicasting applications such as IPTV. To overcome this problem, this paper proposes DCLC model to provide QoS. The DCLC problem aims to minimize the cost of multicast tree while guaranteeing end-to-end multicast tree between source and destinations with respect to predefined delay threshold.

In DCLC problem, the network is considered as a directed graph $G(N, L)$, where N is a set of network nodes and L is a set of network links. In this graph, s is a source node and $V = \{v_1, v_2, \ldots v_n\}$ is a set of destination nodes, T is a multicast tree which connects s node to all destination nodes, $P(s, v)$ is a feasible path from s to v

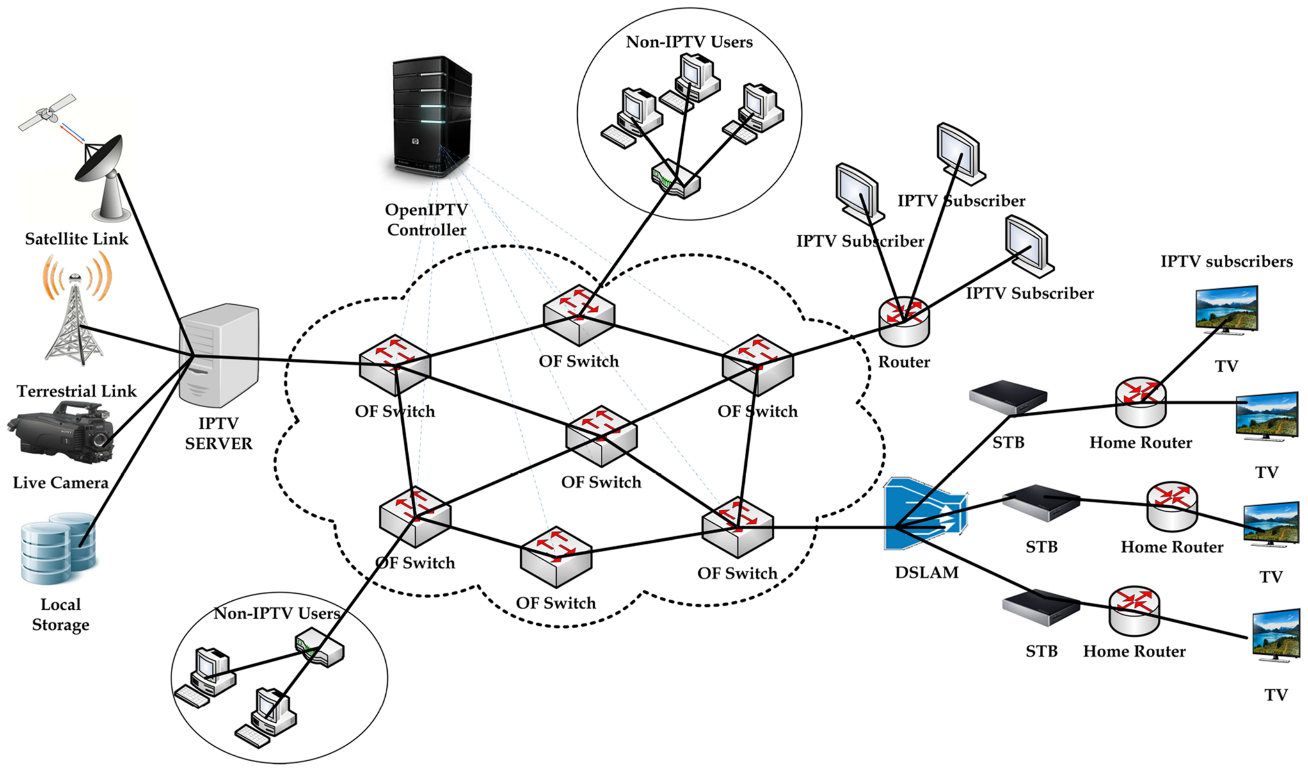


Fig. 1 The architecture of the proposed OpenIPTV framework

and e is a link on the path $P(s, v)$. The minimum cost tree $\forall v \in D,\ P(s,v) \in T$ defines as follows:

$$T^* = \min \sum_{v\in D} \sum_{e\in P(s,v)} \cos t(e). \tag{1}$$

The end-to-end delay of path between s and destination v is the aggregated delay of each link along the path which is defines as follows:

$$\text{Delay } P(s,v) = \sum_{e\in P(s,v)} \text{Delay}(e). \tag{2}$$

The DCLC problem aims to find minimum cost tree between node s and D set as:

$$\begin{aligned} T^* &= \min \sum_{v\in D} \sum_{e\in P(s,v)} \cos t(e) \\ &\text{s.t.}\quad \text{Delay } P(s,v) \le \Delta \\ &\quad \forall v \in D,\quad P(s,v) \in T, \end{aligned} \tag{3}$$

where Δ is the end-to-end delay bound of path between s and v on the minimum multicast tree [28]. Due to NP-completeness of DCLC problem [14], some heuristic and approximation algorithms have been proposed in literatures [28, 29]. This paper utilizes Adaptive Multiple Constraints Routing Algorithm (MAMCRA) [30] for finding efficient multicast tree regarding to end-to-end delay QoS metric. MAMCRA calculates multicast tree between source and destinations and then reduces resource consumption without violating the QoS constraints. In MAMCRA, the set of shortest path between source and all multicast group destinations is calculated. Then, MAMCRA tunes the multicast tree with respect to QoS constraints and finds efficient multicast tree. MAMCRA uses Self-Adapting Multiple Constraints Routing Algorithm (SAMCRA) [31] to construct minimum multicast tree. The length of each path in SAMCRA is nonlinear and is calculated with respect to the weight of each link in terms of constraints values as follows:

$$l(P) = \max_{1\le i\le m}\left(\frac{w_i(P)}{L_i}\right), \tag{4}$$

where m is the number of constraint metrics, L_i is the maximum value of the ith metric and $w_i(P)$ is the vector of the path of the ith metric which is calculated as follows:

$$w_i(P) = \sum_{e\in P} w_i(e), \tag{5}$$

where e is a link on the path P and $w_i(e)$ is the weight of the link e associated to the ith metric. The nonlinearity of path guarantees that a calculated path lies within the constraints, i.e., $l(P) \le 1$. Moreover, in multiple constraints problems, this nonlinearity suggests to consider more paths than only the shortest [31]. For this reason, SAMCRA calculates k-shortest paths using Dijkstra's algorithm. In each step of k-shortest path computation, non-dominant sub paths are selected. A non-dominant path is a path which dominates other paths in terms of all constraint metrics. Finally, SAMCRA returns the minimum non-dominant path that satisfies all constraints as the optimal path. MAMCRA leverages the capabilities of SAMCRA to calculate the shortest path tree between the source and destinations in the network for each of the m link weights separately. In the first step of MAMCRA, the set S of the shortest paths from the source node to each of destination nodes is calculated using SAMCRA. Then, in the second step, the overlaps between the paths in the set S are removed and the shortest path tree is created. Figure 2 shows the flowchart of MAMCRA algorithm. According to this figure, in each iteration, a minimum non-dominant path (Sp) is calculated for a destination node and then added to T. T is a variable to store individual paths from source node to each destination. At the final step of the algorithm, all paths in T are combined to construct a minimum cost multicast tree. This process removes loops and overlaps in a greedy approach and finally the algorithm returns the solution.

In this paper, the QoS constraint for MAMCRA is end-to-end delay between the source of multicast tree and each of destinations. Therefore, MAMCRA finds the minimum cost multicast tree in which the end-to-end delay between the source and each of destinations is less or equal than the delay threshold. MAMCRA can guarantee QoS to the multicast subscribers in an efficient but not always optimal manner. For a single constraint problem such as DCLC, the time complexity of MAMCRA is $O(N \log N + E + Np^2)$,where N is the number of nodes, E is the number of links and p is the number of destinations [30, 31]. Therefore, MAMCRA is a fast and lightweight algorithm for solving DCLC problem and has low overhead.

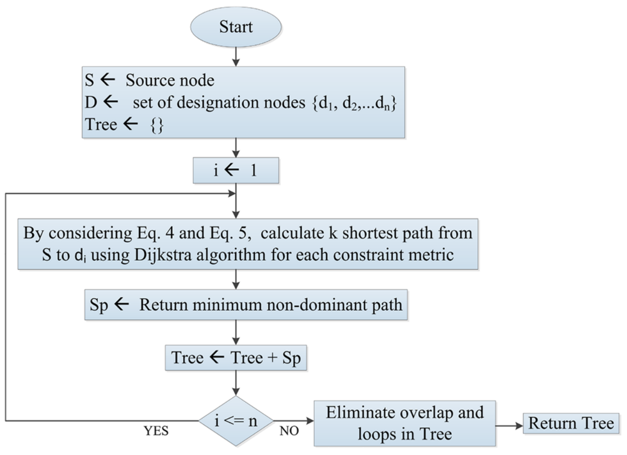

Start
S ← Source node
D ← set of designation nodes {d1, d2,...dn}
Tree ← {}
i ← 1
By considering Eq. 4 and Eq. 5, calculate k shortest path from S to di using Dijkstra algorithm for each constraint metric
Sp ← Return minimum non-dominant path
Tree ← Tree + Sp
i <= n
YES
NO
Eliminate overlap and loops in Tree
Return Tree

Fig. 2 Flowchart of MAMCRA algorithm

3.3 OpenIPTV controller design

As shown in Fig. 3, OpenIPTV controller offers various interfaces and functions to deliver IPTV services.

OpenIPTV controller connects to OF switches with a secure channel using the OF protocol to share necessary information. The controller sends forwarding rules associated with data flows. Moreover, the controller collects network state information from OF switches for discovering the network topology or monitoring the network traffics behavior. In this paper, OpenIPTV controller functions are implemented as a set of modules in the OpenDayLight [32] controller. Modular design of OpenIPTV helps to add new modules and services for future requirements.

According to Fig. 3, OpenIPTV controller includes the following modules:

- Links Available Bandwidth Collector (LABC): frequently acquires information about available bandwidth of each link in network and stores them in the available bandwidth matrix.
- Links Delay Collector (LDC): frequently acquires information about delay of each link in network and stores them in delay matrix.
- Multicast Group Management (MGM): this module is responsible for managing multicast groups and processes IPTV server query management and IPTV subscribers join/leave messages. When the network established, OpenIPTV controller sends forwarding rules to all OF switches in the backbone network and enforces them to forward any multicast group messages include query, join and leave to the OpenIPTV controller. Then, IPTV server sends the list of TV channels to the OpenIPTV controller as a query message. MGM receives the query message and floods it to all IPTV subscribers and creates a separate list for each channel. The flow diagram of this process is depicted in Fig. 4.

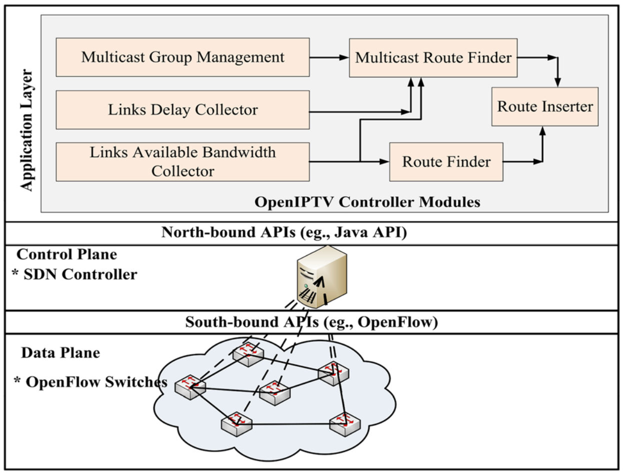
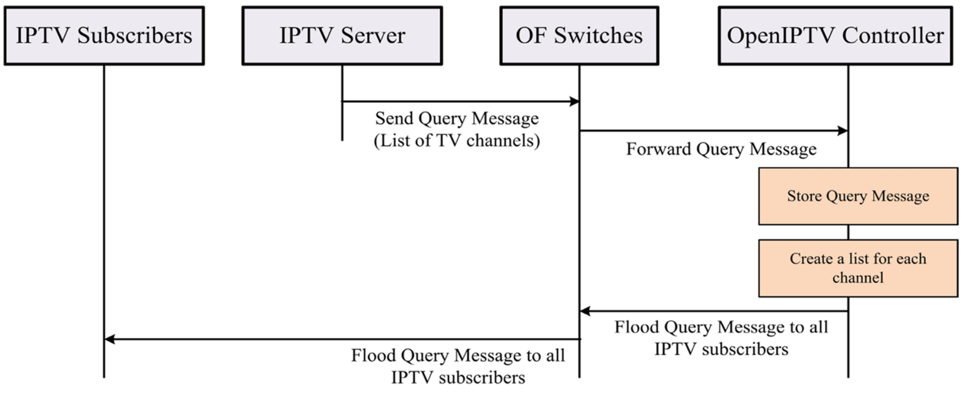

Application Layer
Multicast Group Management
Multicast Route Finder
Links Delay Collector
Route Inserter
Links Available Bandwidth Collector
Route Finder
OpenIPTV Controller Modules
North-bound APIs (eg., Java API)
Control Plane
* SDN Controller
South-bound APIs (eg., OpenFlow)
Data Plane
* OpenFlow Switches

Fig. 3 The proposed OpenIPTV controller

After receiving the query message, each IPTV subscriber can select a channel by sending a join message to the OpenIPTV controller. In an IPTV network with a single IPTV server, each channel corresponds to a multicast group. When a subscriber sends a join message to the controller, MGM stores the IP address of the subscriber who sends the join request in a list which corresponds to the channel which the subscriber requested. It might multiple user watch the same channel in IPTV; hence each list may have multiple user IP addresses. Then, MGM calls MRF module to compute new multicast tree for the subscribers in the list. This mechanism causes considering new subscribers in the multicast tree. Hence, the new subscribers can receive the IPTV contents. Figure 5 shows the flow diagram of joining a user to a channel in OpenIPTV framework.

An IPTV subscriber may intend to exit from a channel. In this case, the subscriber sends a leave message to the controller, MGM checks the corresponding list of the request, deletes the IP address of the subscriber from the list and then calls MRF module to compute new multicast tree for the remained IP addresses in the list. Each IPTV

Fig. 4 Query message flow diagram in proposed OpenIPTV framework


IPTV Subscribers
IPTV Server
OF Switches
OpenIPTV Controller
Send Query Message (List of TV channels)
Forward Query Message
Store Query Message
Create a list for each channel
Flood Query Message to all IPTV subscribers
Flood Query Message to all IPTV subscribers

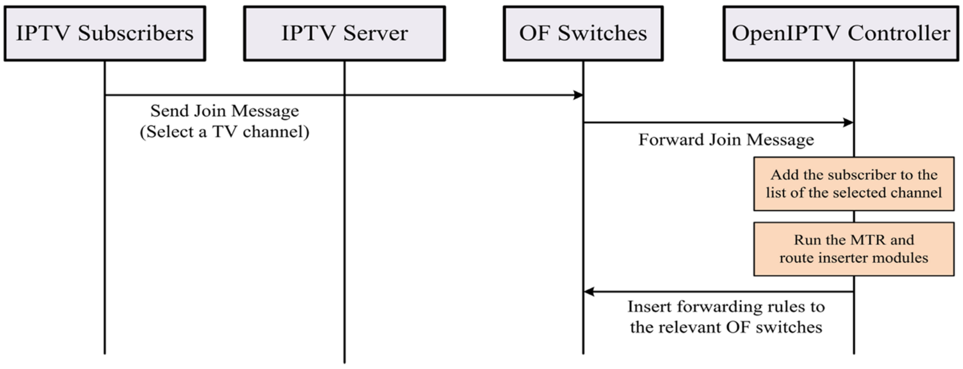

IPTV Subscribers
IPTV Server
OF Switches
OpenIPTV Controller
Send Join Message (Select a TV channel)
Forward Join Message
Add the subscriber to the list of the selected channel
Run the MTR and route inserter modules
Insert forwarding rules to the relevant OF switches

Fig. 5 Join message flow diagram in the proposed OpenIPTV framework

subscriber can send a leave message to exit from a channel and send a join message to re-launch another channel. The flow diagram of leaving from a channel by user is depicted in Fig. 6.

- Multicast Route Finder (MRF): this module periodically executes MAMCRA algorithm to solve the DCLC problem for each multicast group. The MAMCRA algorithm computes the efficient multicast tree based on the available bandwidth and delay of links which have been gathered by the LABC and LDC modules. Because the weight of each link is equal as the available bandwidth of that link, this module chooses the links with high available bandwidth to construct efficient multicast tree. Therefore, this mechanism can diminish packet loss by choosing low-congested links. In summary, this module ameliorates QoS in two aspects: first, it chooses low-congested links and secondly it considers end-to-end delay constrained between the source and destinations. If MAMCRA cannot find any solution that satisfies the end-to-end delay constraint, this module computes minimum cost multicast tree without any constraint. As mentioned before, this module also can be triggered by MGM module when a subscriber leaves or joins from/ to a channel. This module only calculates the efficient multicast trees for IPTV subscribers.
- Route Finder (RF): because there are other non-IPTV users, which use the backbone network, this module calculates shortest path using Dijkstra algorithm for non-IPTV users.
- Route Inserter (RI): this module executes after MRF or RF modules and injects the output of these modules as a set of forwarding rule entries into the relevant switches. Because OpenFlow 1.3 [33] supports group table, this module can define a group table for each multicast group. Each group table contains multiple buckets and each bucket corresponds to a specific port of switch.

4 Performance evaluation

In this section, a comprehensive simulation study is conducted and the results are analyzed to evaluate the performance of OpenIPTV against Multiflow [13], because Multiflow is the only solution that can be implemented in current SDN network without changing the OF switches and messages. As described in Sect. 2, the other proposed solutions are partial solutions and need to change the OF messages formats and OF switch structure. To make a fair comparison, OpenIPTV and Multiflow are evaluated in two different test-bed scenarios as follows:

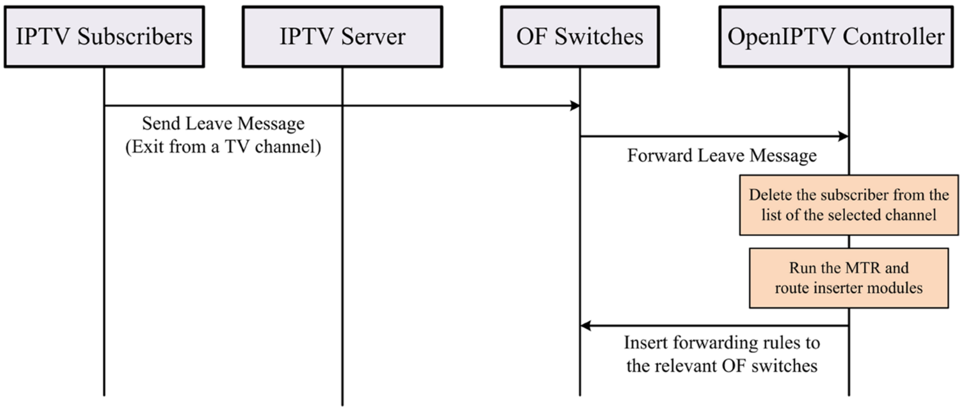

IPTV Subscribers
IPTV Server
OF Switches
OpenIPTV Controller
Send Leave Message (Exit from a TV channel)
Forward Leave Message
Delete the subscriber from the list of the selected channel
Run the MTR and route inserter modules
Insert forwarding rules to the relevant OF switches

Fig. 6 Leave message flow diagram in OpenIPTV

- Scenario-1: in this scenario, the test-bed network consists of 11 OF switches. There are 23 links in the network for connecting OF switches together, and each switch connects to an average 4.18 other switches. The bandwidth and delay of each link is randomly chosen between 10 and 100 Mbps and 30–60 ms, respectively. There are eight IPTV subscribers, which receives IPTV content. Each IPTV subscriber randomly chooses a channel and sends join request to the OpenIPTV controller every 30 s. There are also four non-IPTV users which two of them are senders while the two others are receivers. The non-IPTV users generate and send 3 Mbps UDP traffic to the receivers by Iperf [34].
- Scenario-2: the test-bed network of this scenario consists of 26 OF switches. OF switches are connected together with 50 links and each switch is connected to an average of 3.18 other switches. The bandwidth and delay of each link is randomly chosen between 10 and 100 Mbps and 20–60 ms, respectively. There are 20 IPTV subscribers, which receives IPTV content. Each IPTV subscriber randomly chooses a channel and sends join request to the OpenIPTV controller every 30 s. There are also six non-IPTV users which two of them are senders and the two others are receivers. The non-IPTV users generate and send 3 Mbps UDP traffic to the receivers by Iperf.

4.1 Configuration of simulations

The test-bed networks are implemented on Mininet [35] emulator and the OpenDaylight is used as a network controller. All mentioned OpenIPTV controller are implemented on OpenDayLight. The OpenIPTV controller runs NTM, LABC, LDC, MRF and RI modules every 20 s and announces efficient multicast tree for each multicast group to the relevant switches. Moreover, the MGM module listens to join/leave messages which are sent by IPTV subscribers and manages the multicast groups. Each module of OpenIPTV is developed as a Java class to facilitate modularity. In this way, each module can be called by other modules. We have also implemented Multiflow in OpenDayLight controller according to its specification. The weight of each link of the backbone network is defined as the available bandwidth of that link. In both scenarios, the maximum tolerable end-to-end delay for the MRF module of OpenIPTV controller is set to 200 ms [36]. This value is the Δ threshold value in the DCLC problem which is used for the end-to-end delay constraint in modeling the multicast tree. All experiments run over a machine with CPU Intel Core i7-4700MQ-2.4 GHz and 6 GB RAM. Due to hardware limitation, in this paper the IPTV server can stream ten live different channels over UDP protocol. To assess the performance of OpenIPTV and Multiflow for different workloads, we consider two types of video with different resolutions. The first workload is a H.264 video with resolution 1280×720 at 30 fps, and the second workload is a H.264 video with resolution 1920×1080 at 30 fps. The simulation time for every test is 3 min (180 s). Simulation results are analyzed for different number of IPTV channels.

4.2 Performance metrics

The performance of OpenIPTV and Multiflow is evaluated using the following QoS parameters in both scenarios:

- PSNR (Peak Signal-To-Noise Ratio): PSNR is a useful metric for measuring video quality. This metric is used to calculate the error between the original streamed video and the received video.
- Average end-to-end delay: this metric is the average time interval from the IPTV server to the IPTV subscriber for each successfully delivered video packet.
- Packet loss ratio: packet loss ratio is the total number of video packets, which have not been delivered at the IPTV subscribers to the total number of packets streamed at the IPTV server.
- Pre-roll delay: this metric denotes the time between an IPTV subscriber sending a join request and receiving the first video packet of the selected channel.

4.3 Comparative study

In this section, the simulation results are analyzed for two test-bed scenarios under different number of IPTV channels in terms of QoS metrics. The horizontal axes of all figures in this section show the number of IPTV channels, which the IPTV server streams over the backbone network. Figure 7 illustrates the average end-to-end delay for the test-bed scenarios with different workloads.

As illustrated in this figure, OpenIPTV outperforms Multiflow for different workloads and scenarios in terms of average end-to-end delay. This is because OpenIPTV finds an efficient multicast tree, which satisfies end-to-end delay constraint for each IPTV subscribers. As the matter of fact, OpenIPTV guarantees that the maximum end-to-end delay of each video packet is 200 ms which leads to reduce average end-to-end delay. On the other hand, Multiflow does not care about delay and only finds the shortest multicast tree without any consideration of QoS metrics. In addition, unlike Multiflow, OpenIPTV has a separate module (LDC), which periodically measures and collects current delay of links and the MRF module calculates the multicast tree with respect to the current delay of links. This mechanism causes that in each period, the links with low delay are chosen to construct the multicast tree. From Fig. 7, it can

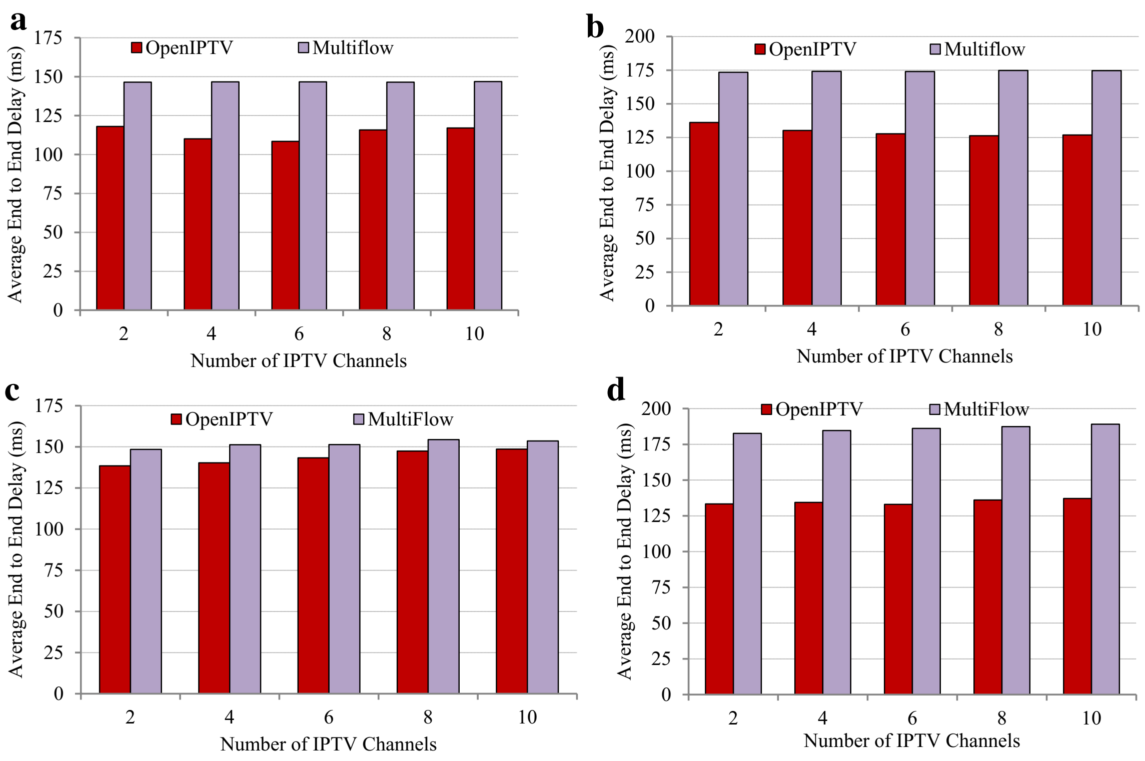


Fig. 7 Average end-to-end delay: **a** test-bed scenario-1 (video 1280 × 720), **b** test-bed scenario-2 (video 1280 × 720), **c** test-bed scenario-1 (video 1920 × 1080), **d** test-bed scenario-2 (video 1920 × 1080 × 720)

be seen that the difference between Multiflow and OpenIPTV increases as the size of network increases which indicates OpenIPTV is more scalable than Multiflow in terms of reducing end-to-end delay. The simulation results for packet loss ratio are depicted in Fig. 8.

As demonstrated in Fig. 8, OpenIPTV shows better performance in terms of packet loss for different workloads and network sizes. Because Multiflow does not consider available bandwidth of links as a weight for links, it might choose congested links for construction of multicast tree. This behavior increases the probability of packet loss. On the other hand, OpenIPTV chooses low-congested links for calculating multicast tree. Moreover, OpenIPTV dynamically gathers the available bandwidth of links using the LABC module. Higher available bandwidth of a link leads to lower congestion, i.e., lower packet loss ratio on the link. For this reason, MRF module assigns the weight of each link based on its available bandwidth. In this way, the MRF finds efficient multicast tree which contains low-congested (high available bandwidth) links and as a result reduces the packet loss ratio.

Figure 9 depicts the results of average PSNR which is the most widely used metric for evaluating the video quality. As we mentioned before, end-to-end delay and packet loss are two important factors that affects the quality of received video. In fact, long end-to-end delay and high packet loss ratio have ill-effect on PSNR and causes lower values for PSNR. According to the Fig. 9, OpenIPTV considerably outperforms Multiflow in terms of average PSNR for different number of IPTV channels and workloads. As mentioned above, because OpenIPTV reduces the packet loss ratio and end-to-end delay for video streams, it can also increase the PSNR of the received IPTV contents. Moreover, as shown in Figs. 7 and 8, Multiflow has worse performance than OpenIPTV in terms of both average end-to-end delay and packet loss which results lower PSNR, i.e., lower quality for received video.

Unlike the previous results for QoS metrics, Fig. 10 shows that Multiflow outperforms OpenIPTV in terms of the pre-roll delay in both network sizes and different workloads. This is because of Multiflow does not gather network statistics and only executes Dijkstra algorithm to find the shortest multicast tree. Therefore, it immediately calculates multicast tree and inserts the forwarding rules to the relevant switches. On the other hand, in OpenIPTV, the controller first gathers network state information including available bandwidth and delay of each link in the network and then solves DCLC problem using the MAMCRA algorithm. In addition to this behavior, the time complexity of MAMCRA algorithm is greater than

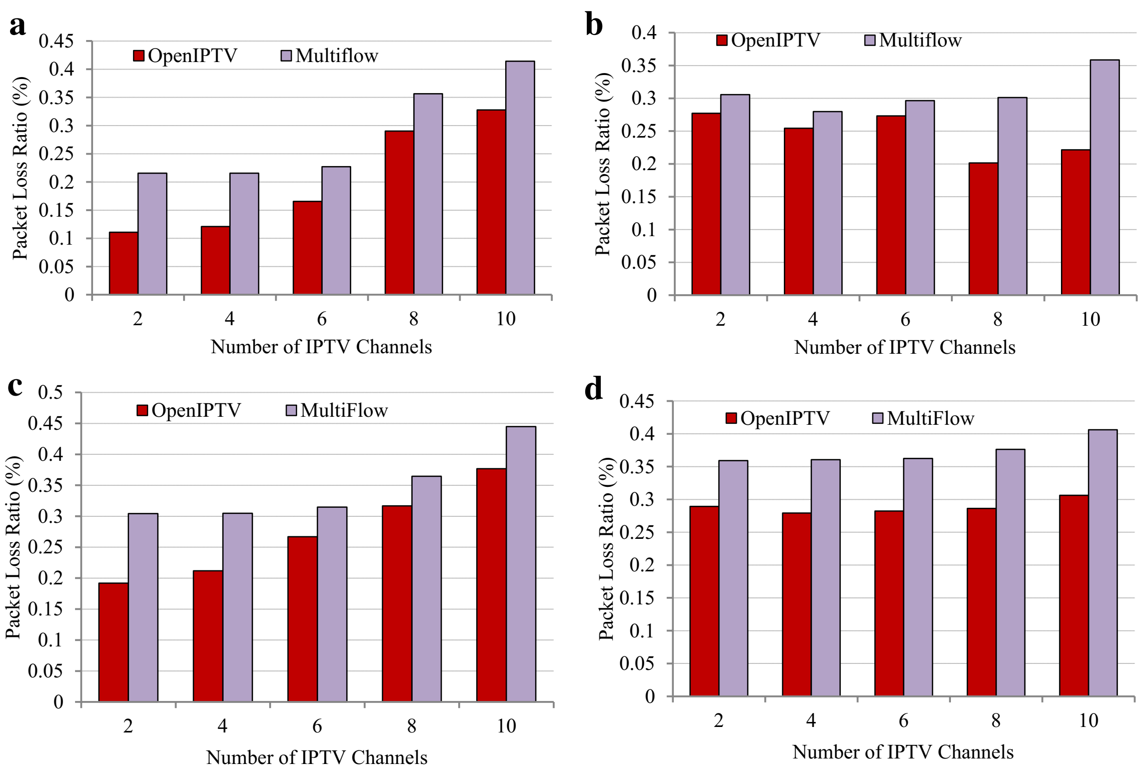


Fig. 8 Packet loss ratio: **a** test-bed scenario-1 (video 1280 × 720), **b** test-bed scenario-2 (video 1280 × 720), **c** test-bed scenario-1 (video 1920 × 1080), **d** test-bed scenario-2 (video 1920 × 1080 × 720)

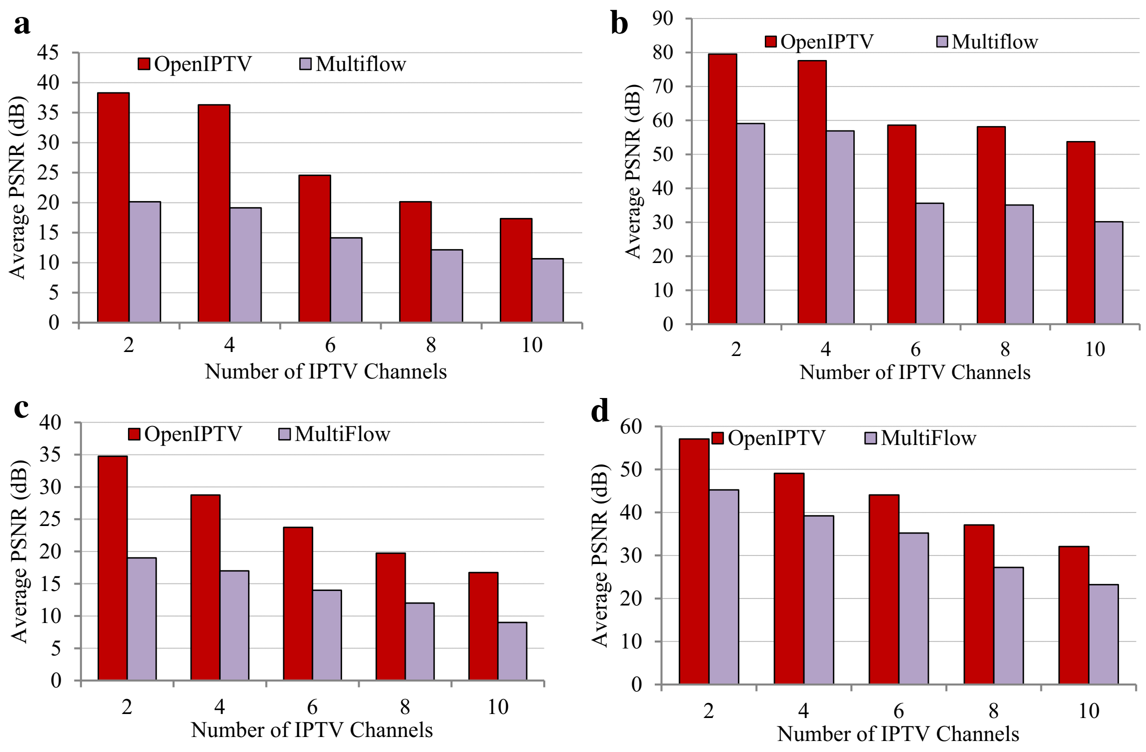


Fig. 9 Average PSNR: **a** test-bed scenario-1 (video 1280 × 720), **b** test-bed scenario-2 (video 1280 × 720), **c** test-bed scenario-1 (video 1920 × 1080), **d** test-bed scenario-2 (video 1920 × 1080 × 720)

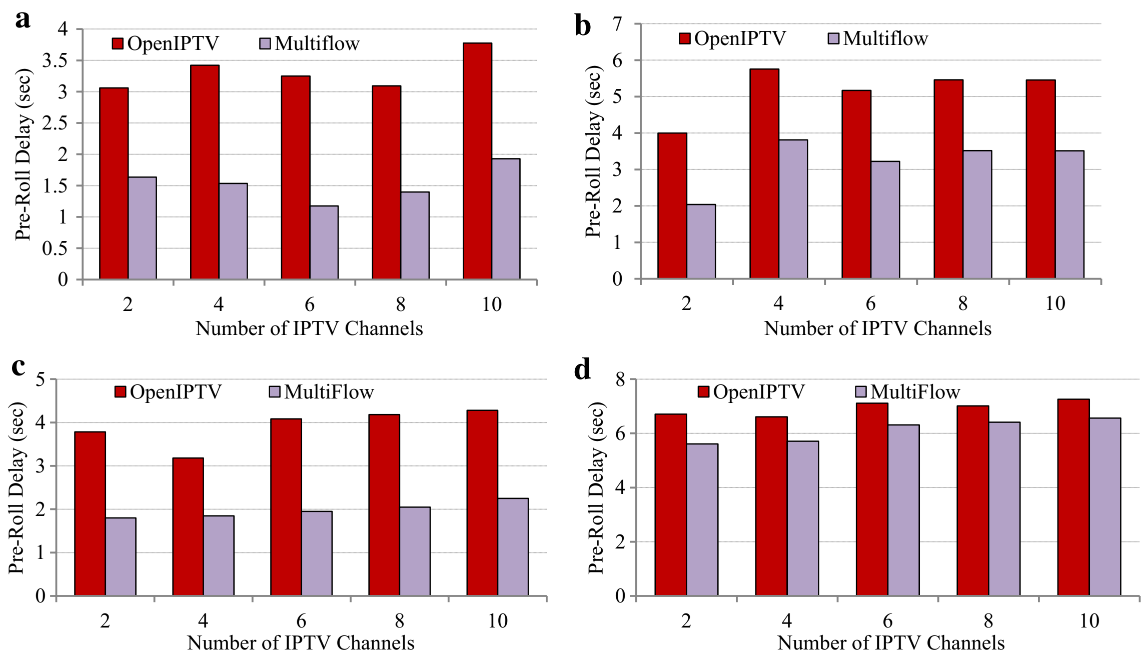


Fig. 10 Pre-roll delay: **a** test-bed scenario-1 (video 1280 × 720), **b** test-bed scenario-2 (video 1280 × 720), **c** test-bed scenario-1 (video 1920 × 1080), **d** test-bed scenario-2 (video 1920 × 1080 × 720)

Dijkstra that causes the controller consumes more time in calculation of multicast tree in OpenIPTV. Therefore, this approach takes more time in compared to Multiflow and cause higher value for the pre-roll delay metric. According to Fig. 10, the only drawback of OpenIPTV is high pre-roll delay.

Table 1 shows the improvements of OpenIPTV compared to Multiflow in terms of packet loss ratio, average PSNR and average end-to-end delay for different workloads and scenarios. The content of Table 1 is a summarization of all simulation results which have been illustrated from Figs. 7, 8 and 9.

5 Conclusions

In this paper, a novel SDN-based IPTV service framework named OpenIPTV is proposed to provide QoS for IPTV contents. OpenIPTV utilizes current OpenFlow equipment as an underlying technology and improves IPTV service delivery with respect to required QoS metrics for IPTV contents. OpenIPTV architecture can deliver IPTV service over a shared backbone network for IPTV subscribers while other non-IPTV subscriber can also use the backbone network for sending their own traffics. In the proposed architecture, OpenIPTV controller acts as a backbone

Table 1 Comparison between OpenIPTV and Multiflow

Video resolution	Scenarios	Number of IPTV channels	2 (%)	4 (%)	6 (%)	8 (%)	10 (%)
1280 × 720	Scenario 1	Average end-to-end delay	24	33	35	26	25
		Average PSNR	90	88	73	65	62
		Packet loss	94	78	37	22	26
	Scenario 2	Average end-to-end delay	27	33	36	38	37
		Average PSNR	26	27	39	40	44
		Packet loss	10	9	8	49	60
1920 × 1080	Scenario 1	Average end-to-end delay	7	8	5	5	3
		Average PSNR	82	68	69	64	85
		Packet loss	58	43	17	15	18
	Scenario 2	Average end-to-end delay	36	37	39	37	37
		Average PSNR	26	25	25	36	38
		Packet loss	24	29	28	31	32

network controller for monitoring and configuring network. OpenIPTV controller is well-designed and modular; makes possible to add future requirements. OpenIPTV controller consists of some modules which manage multicast groups and compute efficient multicast tree between IPTV server and IPTV subscribers. To measure the advantages of OpenIPTV, different scenarios have been explored. Because end-to-end delay, PSNR, pre-roll delay and packet loss ratio are the most important QoS metrics for IPTV service, this paper investigates these metrics for comparison OpenIPTV with the other well-known solutions such as Multiflow. Simulation results demonstrate that OpenIPTV outperforms Multiflow. Furthermore, simulation results show that OpenIPTV is an effective way to deliver IPTV services with high level of QoS over SDN and can be readily used in practice.

References

1. Cha, M., Choudhury, G., Yates, J., Shaikh, A., Moon, S.B.: Case study: resilient backbone design for IPTV services. In: Workshop on IPTV Services over World Wide Web held in conjunction with WWW2006. ACM (2006)
2. Noghani, K.A., Oguz Sunay, M.: Streaming multicast video over software-defined networks. In: 2014 IEEE 11th International Conference on Mobile Ad Hoc and Sensor Systems, pp. 551–556 (2014)
3. http://www.iptvmagazine.com/stats.html. Accessed June 2016 (2016)
4. Xiao, Y., Du, X., Zhang, J.: Internet protocol television (IPTV): the killer application for the next-generation internet. In: Institute of Electrical and Electronics Engineers (2007)
5. Punchihewa, A., De Silva, A.M.: Tutorial on IPTV and its latest developments. In: 2010 Fifth International Conference on Information and Automation for Sustainability, pp. 45–50. IEEE, New York (2010)
6. Sokolov, I., Belyaev, L., Baronshin, I., Kirakosyan, A.: Dynamic channel selection for live and previously broadcast content. In: U.S. Patent 20,160,150,284, issued 26 May 2016 (2016)
7. Kuo, JW.: Multiple device IPTV cloud-based recording and playback. In: U.S. Patent Application 13/018,566, filed 2 Aug 2012 (2012)
8. Fenner, B., He, H., Haberman, B., Sandick, H.: Internet Group Management Protocol (IGMP)/Multicast Listener Discovery (MLD)-Based Multicast Forwarding, RFC 4605 (2006)
9. Moy, J.: Multicast extension to ospf: RFC 1584 (1994)
10. Farinacci, D., Liu, C., Deering, S., Estrin, D., Handley, M., Van Jacobson, L., Wei, P.S., Thaler, D., Helmy, A.: Protocol Independent Multicast-Sparse Mode (PIM-SM), RFC2362 (1998)
11. Pusateri, T.: DVMRP Version 3. Juniper Networks (2001) (**work in progress**)
12. Ballardie, T., Francis, P., Crowcroft, J.: Core based trees (CBT). In: ACM SIGCOMM Computer Communication Review, vol. 23(4), pp. 85–95. ACM, London (1993)
13. Bondan, L., Müller, L.F., Kist, M.: Multiflow: multicast clean-slate with anticipated route calculation on OpenFlow programmable networks. J. Appl. Comput. Res. **2**(2), 68–74 (2013)
14. Neto, A., Cerqueira, E., Curado, M., Monteiro, E., Mendes, P.: Scalable resource provisioning for multi-user communications in next generation networks. In: Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008, pp. 1–6. IEEE, New York (2008)
15. Akyildiz, I.F., Ahyoung Lee, P., Wang, M.L., Chou, W.: A roadmap for traffic engineering in SDN-OpenFlow networks. Comput. Netw. **71**, 1–30 (2014)
16. Goransson, P., Black, C.: Software Defined Networks: A Comprehensive Approach. Elsevier, New York (2014)
17. Thorpe, C., Olariu, C., Hava, A., McDonagh, P.: Experience of developing an openflow SDN prototype for managing IPTV networks. In: 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM), pp. 966–971. IEEE, New York (2015)
18. McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., Turner, J.: OpenFlow: enabling innovation in campus networks. ACM SIGCOMM Comput. Commun. Rev. **38**(2), 69–74 (2008)
19. Sriram, R., Manimaran, G., Siva Ram Murthy, C.: Preferred link based delay-constrained least-cost routing in wide area networks. Comput. Commun. **21**(18), 1655–1669 (1998)
20. Zhou, L.: On data-driven delay estimation for media cloud. IEEE Trans. Multimed. **18**(5), 905–915 (2016)
21. Zhou, L., et al.: Energy-spectrum efficiency tradeoff for video streaming over mobile ad hoc networks. IEEE J. Sel. Areas Commun. **31.5**, 981–991 (2013)
22. Mohammadi, R., Javidan, R.: An adaptive type-2 fuzzy traffic engineering method for video surveillance systems over software defined networks. Multimed. Tools Appl. (2016). doi:10.1007/s11042-016-4137-0
23. Patrick, M., Olariu, C., Hava, A., Thorpe, C.: Enabling IPTV service assurance using openflow. In: 2013 27th International Conference on Advanced Information Networking and Applications Workshops (WAINA), pp. 1456–1460. IEEE, New York (2013)
24. Marcondes, C.A.C., Santos, T.P.C., Godoy, A.P., Viel, C.C., Teixeira, C.A.C.: CastFlow: clean-slate multicast approach using in-advance path processing in programmable networks. In: 2012 IEEE Symposium on Computers and Communications (ISCC), pp. 000094–000101. IEEE, New York (2012)
25. Rattanawadee, P., Ruengsakulrach, N., Saivichit, C.: The transmission time analysis of IPTV multicast service in SDN/OpenFlow environments. In: 2015 12th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), pp. 1–5. IEEE, New York (2015)
26. Kotani, D., Suzuki, K., Shimonishi, H.: A design and implementation of openflow controller handling IP multicast with fast tree switching. In: 2012 IEEE/IPSJ 12th International Symposium on Applications and the Internet (SAINT). IEEE, New York, pp. 60–67 (2012)
27. Coras, F., Domingo-Pascual, J., Maino, F., Farinacci, D., Cabellos-Aparicio, A.: Lcast: software-defined inter-domain multicast. Comput. Netw. **59**, 153–170 (2014)
28. Ling, Z., Wei-xiong, D., Yu-xi, Z.: Delay-Constrained Multicast Routing Algorithm Based on Average Distance Heuristic. arXiv:1003.3317 (2010) (**arXiv preprint**)
29. Xu, Y.: Metaheuristic Approaches for QoS Multicast Routing Problems. University of Nottingham, Nottingham (2011)
30. Kuipers, F., Van Mieghem, P.: MAMCRA: a constrained-based multicast routing algorithm. Comput. Commun. **25**(8), 802–811 (2002)
31. Van Mieghem, P., Kuipers, F.A.: On the complexity of QoS routing. Comput. Commun. **26**(4), 376–387 (2003)
32. https://www.opendaylight.org. Accessed 17 May 2016 (2016)
33. Specification, OpenFlow Switch "V1. 3.1.". https://www.opennetworking.org/images/stories/downloads/sdn-resources/

onf-specifications/openflow/openflow-spec-v1.3.0.pdf. Accessed 17 May 2016 (2012)
34. Tirumala, A., Qin, F., Dugan, J., Ferguson, J., Gibbs, K.: Iperf: the TCP/UDP Bandwidth Measurement Tool. Available from: http://sourceforge.net/projects/iperf/. Accessed 17 May 2016 (2005)
35. http://mininet.org/. Accessed 17 May 2016 (2016)
36. Egilmez Hilmi, E., Civanlar, S., Murat Tekalp, A.: An optimization framework for QoS-enabled adaptive video streaming over OpenFlow networks. In: IEEE Transactions on Multimedia, vol. 15(3), pp. 710–715 (2013)