A Load-Balanced Video Multicast Routing System in Software-Defined Networks

Meng-Wei Lee 李孟葳

2015/10

Outline

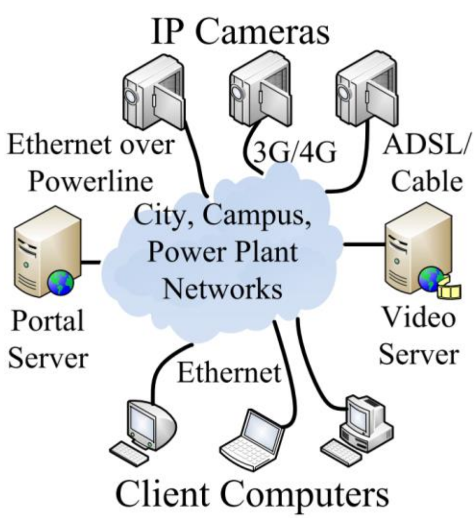
- **Motivation**
- System Overview
- Problem Formulations & Proposed Solutions
- Testbed
- Experiments in Mininet
- Conclusion

Motivation

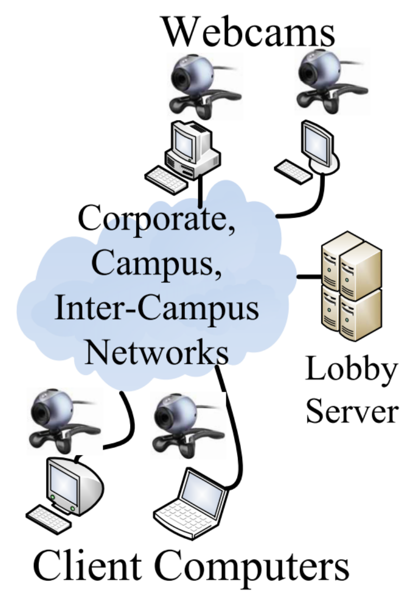
- Video streams dominate the Internet traffic:
  - IP video traffic will be 80% of all IP traffic by 2019, up from 67% in 2014 [1]
- Usage scenarios

  (a) Video surveillance networks

  (b) Video conferencing services



(a)



(b)

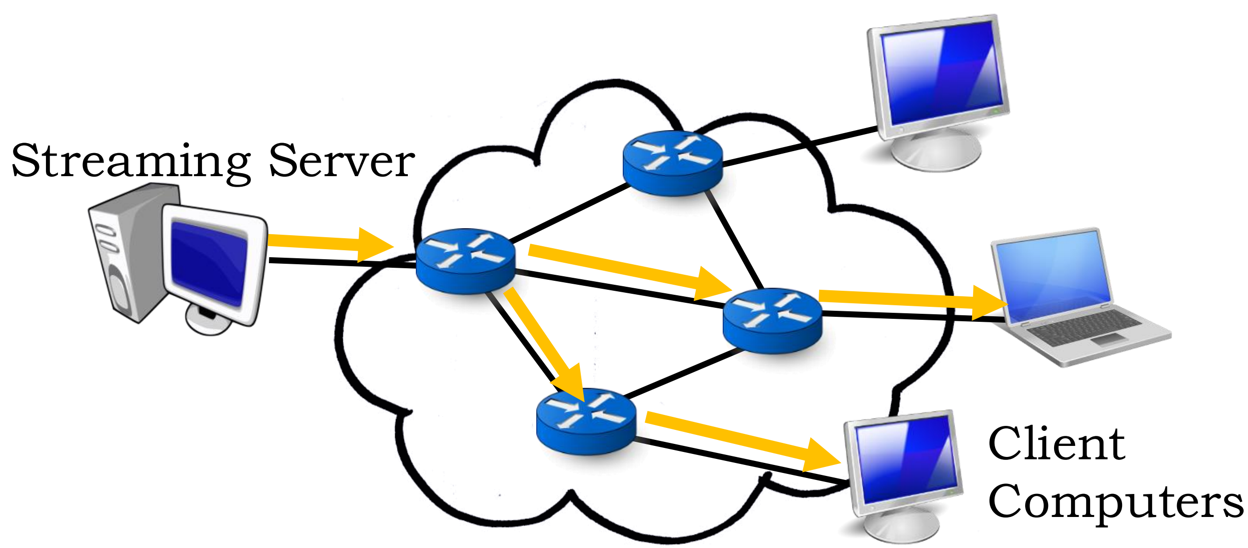
[1] Cisco visual networking index: Forecast and methodology, 2014-2019

Motivation (cont.)

- Real-time streaming imposes strict Quality of Service
  - Requirements on connections of latency, packet loss rate, and bandwidth …
- **How to reduce redundant streams and video bandwidth requirements?**

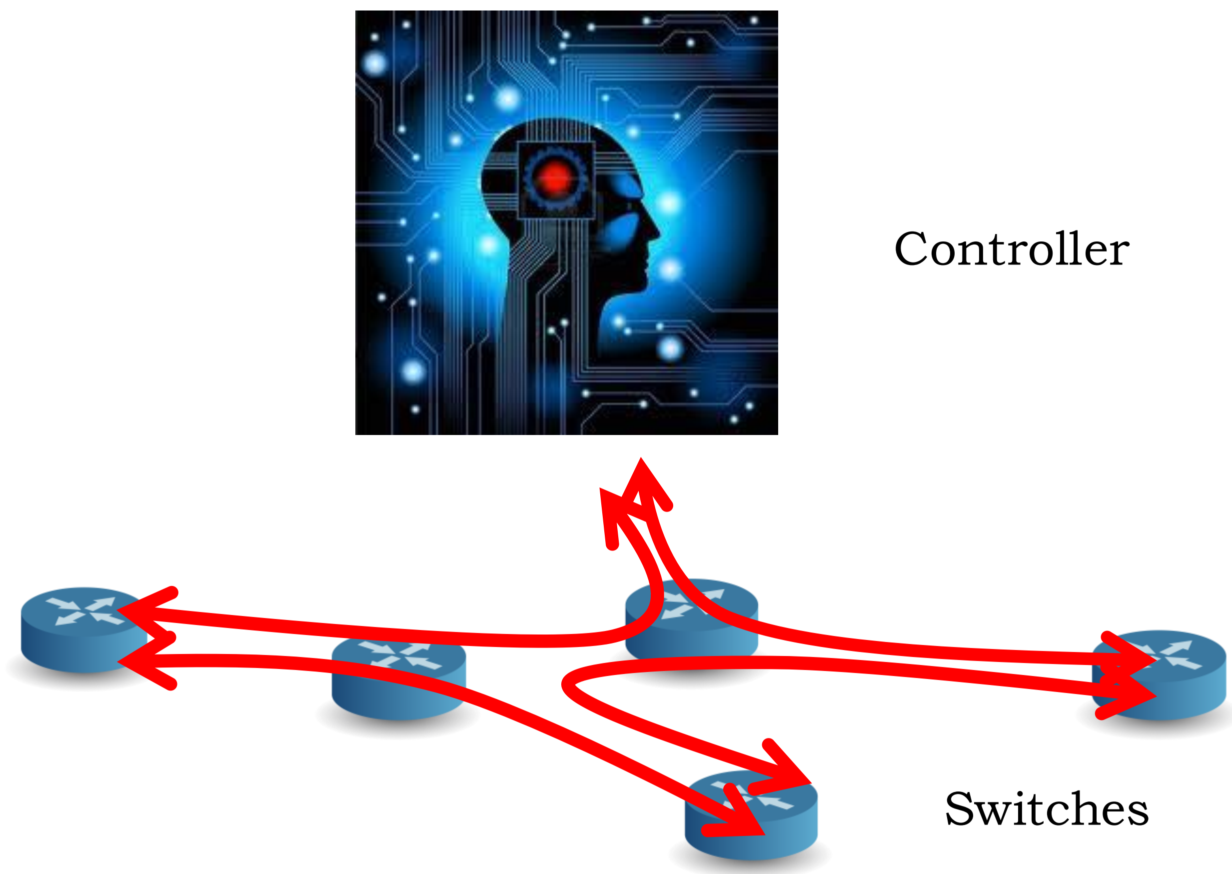
Current Solution

- IP multicast
  - Complex operation – IGMP, PIM, MOSPF, and DVMRP
  - Need costly devices



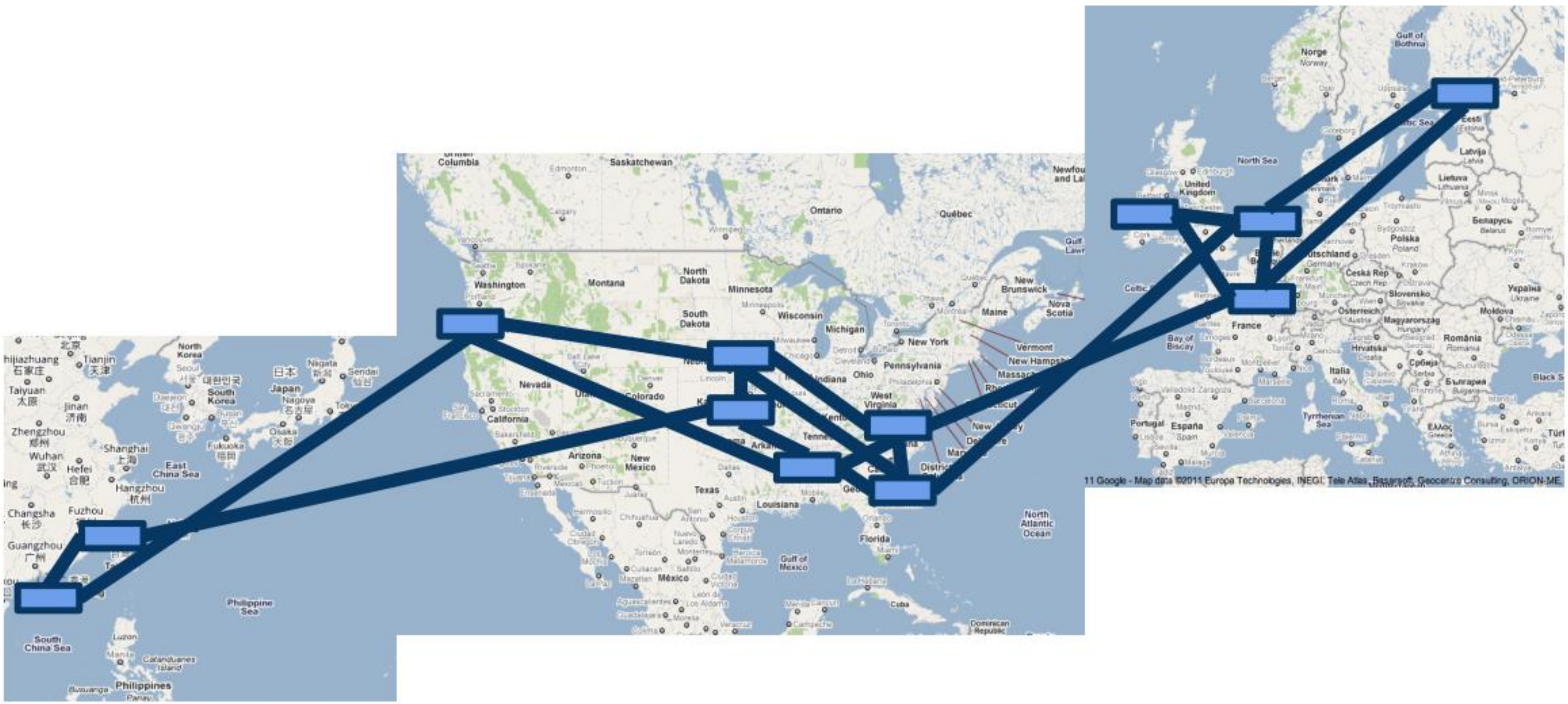
Our Solution

- Software-Defined Networks (SDNs)



Successful Case

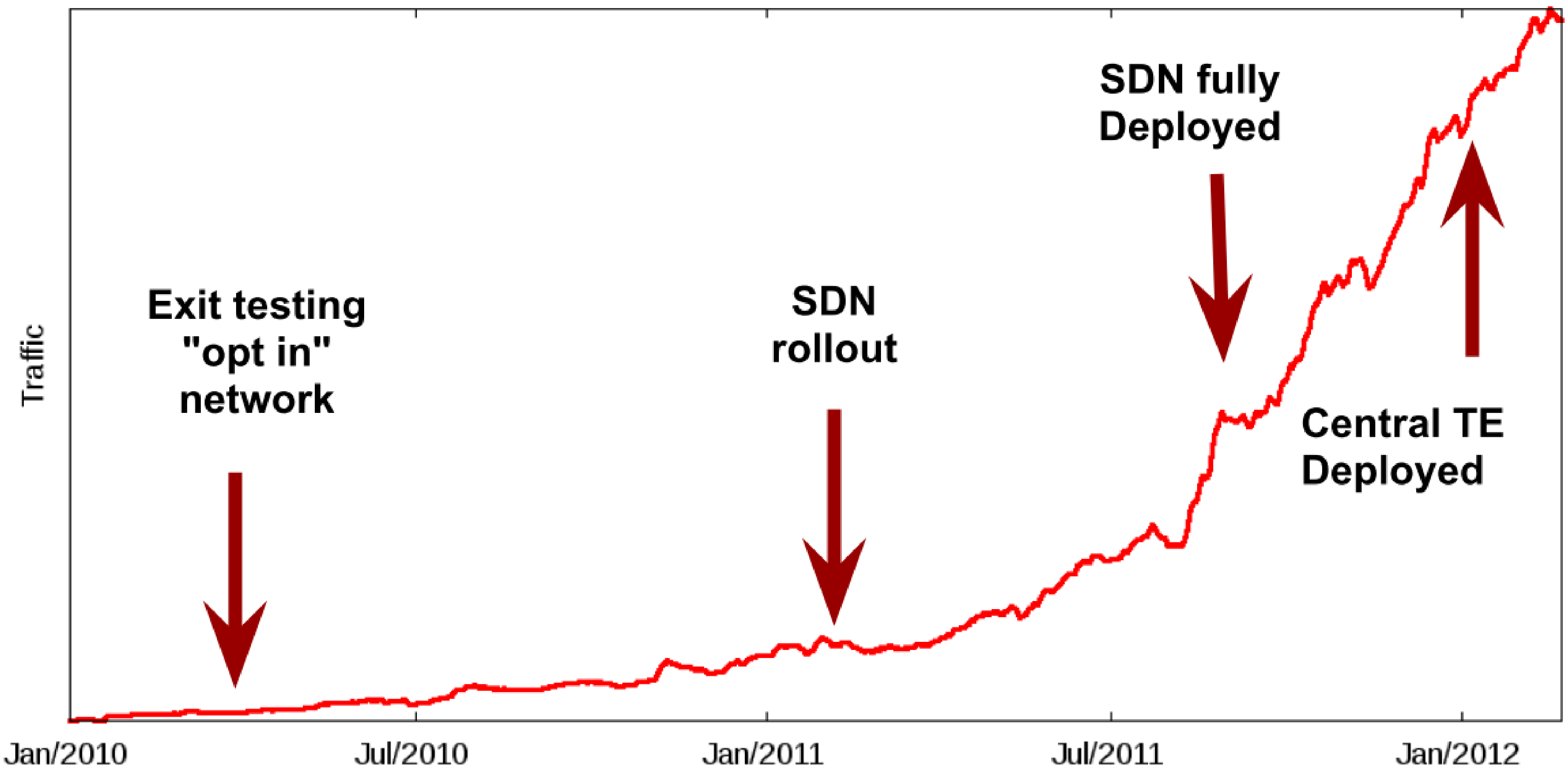
- Google's SDN (OpenFlow) WAN (2010 - 2012)



Source: Google

Successful Case

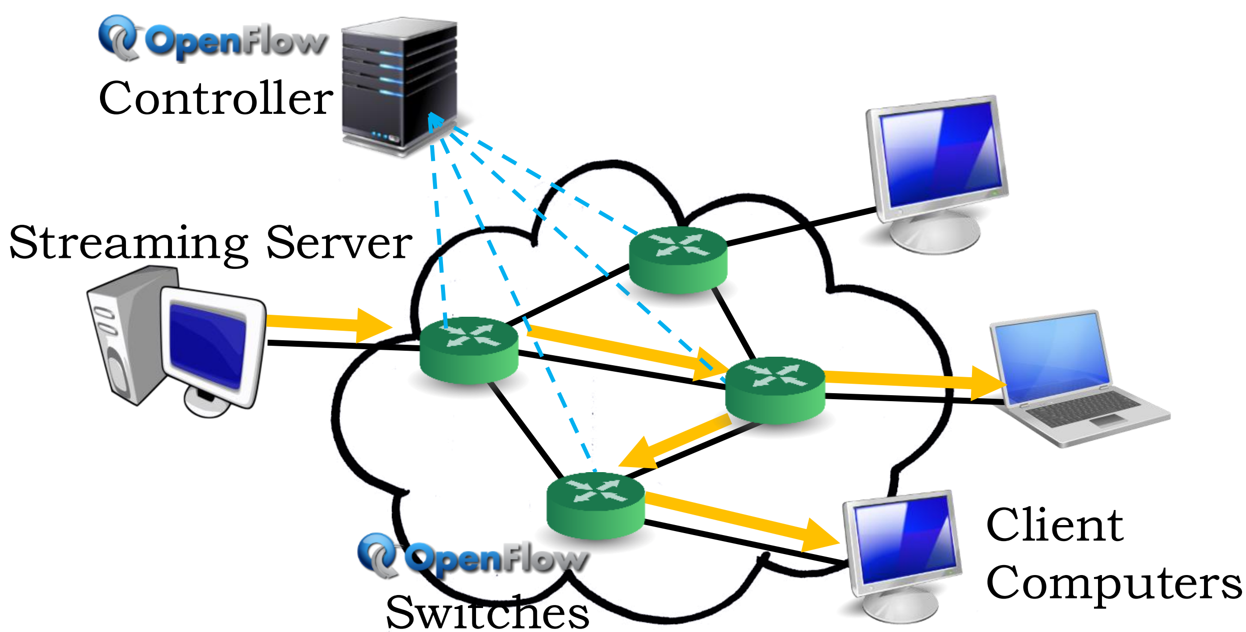
- Google's SDN (OpenFlow) WAN (2010 - 2012)



Source: Google

Benefits of Using SDNs

- Centralized multicast routing algorithm
- Lower network overhead
- Lower setup cost and maintenance cost



Related Work

- **Multicast in SDNs**
  1. [37] proposes a multi-party video conferencing system
  2. [20] focuses on customizing the multicast services
  - They only discuss the performance of existing multicast routing algorithms

  - We design the **centralized multicast routing algorithms**, which aims to **balance the load** among the network links
  - Utilize **Multiple Description Coding (MDC)** for **robust multipath** multicast routing

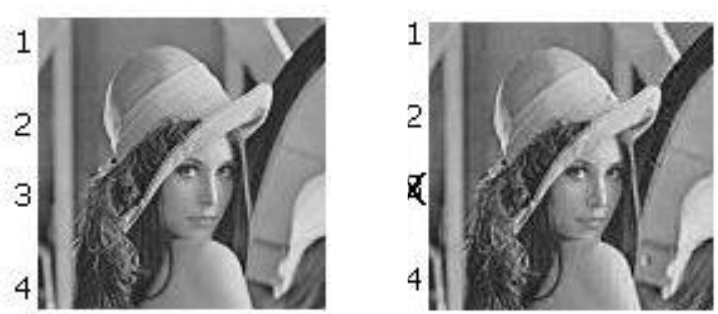
[37] M. Zhao, B. Jia, M. Wu, H. Yu, and Y. Xu., "Software defined network-enabled multicast for multi-party video conferencing systems," *in Proc. of IEEE International Conference on Communications (ICC'14),* Sydney, Australia, June 2014

[20] S. Liao, X. Hong, C. Wu, B. Wang, and M. Jiang. "Prototype for customized multicast services in software defined networks," In *Proc. of IEEE Software, Telecommunications and Computer Networks (SoftCOM'14),* Split, Croatian, September 2014

Multiple Description Coding (MDC)

1. Separate a frame into several descriptors
2. Any subset of descriptors can be decoded
3. Transmitted over separate channels

- Provide **error resilience** to media streams



(a) (b)

An image transmitted in 4 descriptors by MDC
(a) 4 descriptors received
(b) Descriptor #3 is lost
Quality improves while more descriptors received

BARNETT, Eran; KASPI, Yoni. Multiple Description Image Coding. 2005.

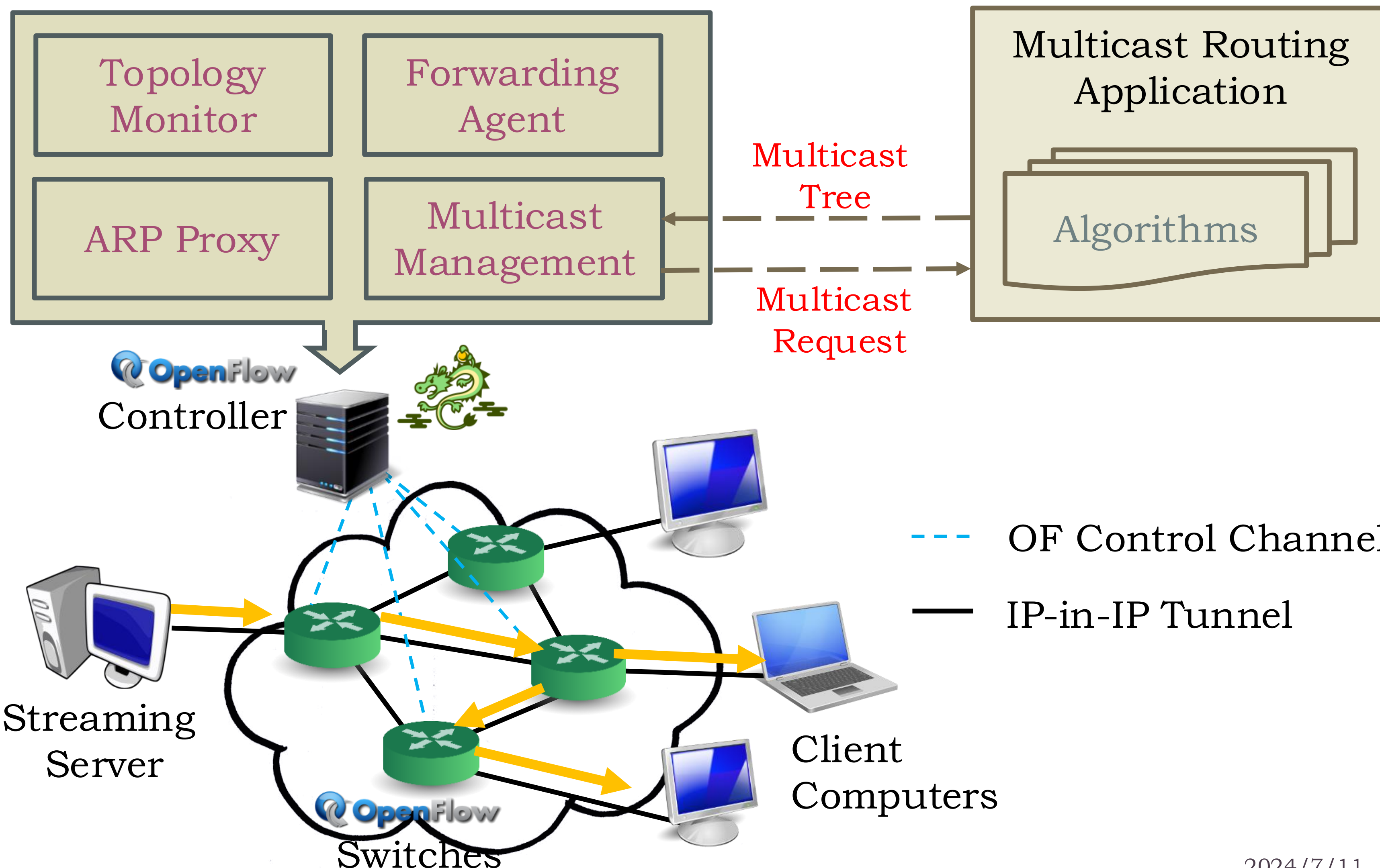
Design Objectives

- Multipath multicast routing system in SDNs
  - Robustness
  - Load Balancing
  - Adaptation
  - Reliability
  - SDN Compatible

Outline

- Motivation
- **System Overview**
- Problem Formulations & Proposed Solutions
- Testbed
- Experiments in Mininet
- Conclusion

System Overview



Outline

- Motivation
- System Overview
- **Problem Formulations & Proposed Solutions**
- Testbed
- Experiments in Mininet
- Conclusion

Problem Formulation - Objective

- We consider the network topology as a directed graph

$$G = (V, E)$$

A switch, a source, or a sink — A network link between two vertices

Objective:

Minimize the maximal link utilization

$$\underset{1 \le e \le E}{\text{minimize} \quad \max} \sum_{s=1}^{S} \frac{R_s}{K} \sum_{k=1}^{K} \frac{x_{\epsilon_e, \varsigma_e}^{s,k}}{b_e} \qquad (1)$$

x=1, if the link is one of the multicast tree edge

bandwidth of the link

link utilization

S: # video sources
K: # video descriptors
R_s: bit rate of video s

Problem Formulation – Constraints

s.t.

1. For each vertex

- Limit the number of flow-entries

$$\sum_{s=1}^{S}\sum_{k=1}^{K}\sum_{e\in O_v} x_{\epsilon_e,\varsigma_e}^{s,k} \le f_v\,, \forall v \in [1,V] \quad (2)$$

2. For each edge

- The traffic flow must be less than its bandwidth

$$\sum_{s=1}^{S}\frac{R_s}{K}\sum_{k=1}^{K} x_{\epsilon_e,\varsigma_e}^{s,k} \le b_e\,, \forall e \in [1,E] \quad (3)$$

3. For each video

- To be robust, a switch can participate in up to $K-1$ multicast trees

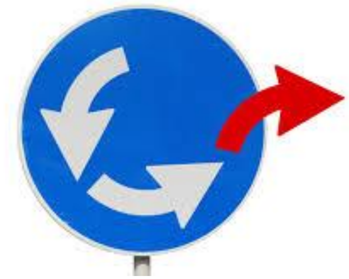
$$\sum_{k=1}^{K}\sum_{e\in I_v} x_{\epsilon_e,v}^{s,k} \le K-1\,, \forall s \in [1,S], v \in [1,V] \quad (4)$$

Robust Multipath Multicast Routing

- It is an Integer Programming (IP) problem
- We use IBM CPLEX to solve the optimal solution of network load balancing

- The solution may contain cycles

Cycle Elimination



- Whenever the cycle occurs
  - Add more constraints to the formulation
  - Save more time rather than prevent cycle in advance

➢ **Optimal solution of multipath multicast trees**

- Abbreviated as ***RMMR****
  - Optimal **R**obust **M**ultipath **M**ulticast **R**outing Algorithm

- But it is still time consuming in large networks
  - **Need a more efficient solution!**

Route Adaptation Heuristics

- Only modify a part of multicast trees for sinks join/leave

Heuristic:

Sink joins ➡

function JOIN(video sink c, video s)
  for all descriptor $k = 1,2,\ldots K$ **do**
    perform BFS from c for the shortest path to tree k
    add the shortest path with minimal link utilization to tree k

Sink leaves ➡

function LEAVE(video sink c, video s)
  for all descriptor $k = 1,2,\ldots K$ **do**
    loop switches from c to the root of tree k
    reduce the count of down-stream video sinks
    remove the switch with 0 count from tree k

- **RMMR:** Efficient Robust Multipath Multicast Routing Algorithm
  - Call RMMR* for optimal multicast trees first
  - Run above heuristics to update multicast trees

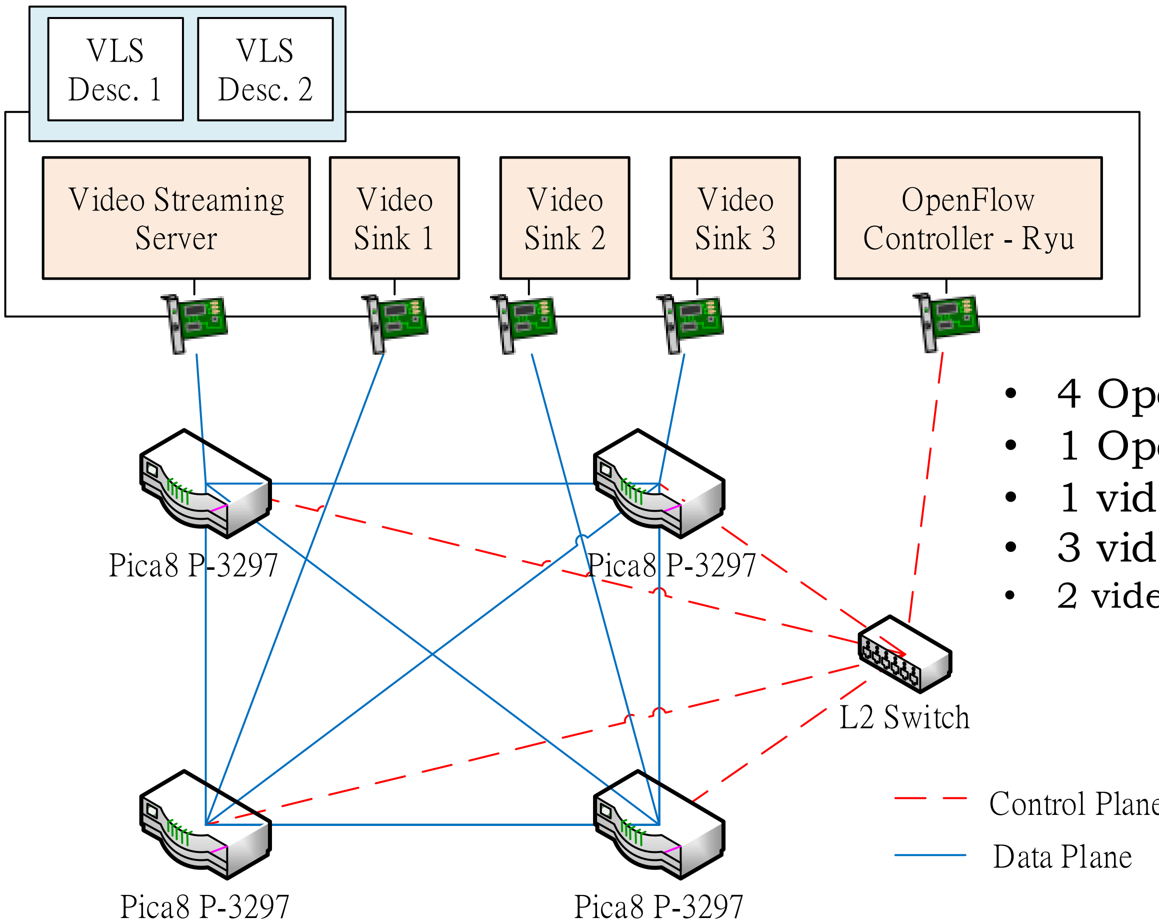
Robust Multipath Multicast Routing Algorithms

- **RMMR*** 
  - Optimal Robust Multipath Multicast Routing Algorithm
- **RMMR** 
  - Efficient Robust Multipath Multicast Routing Algorithm

Outline

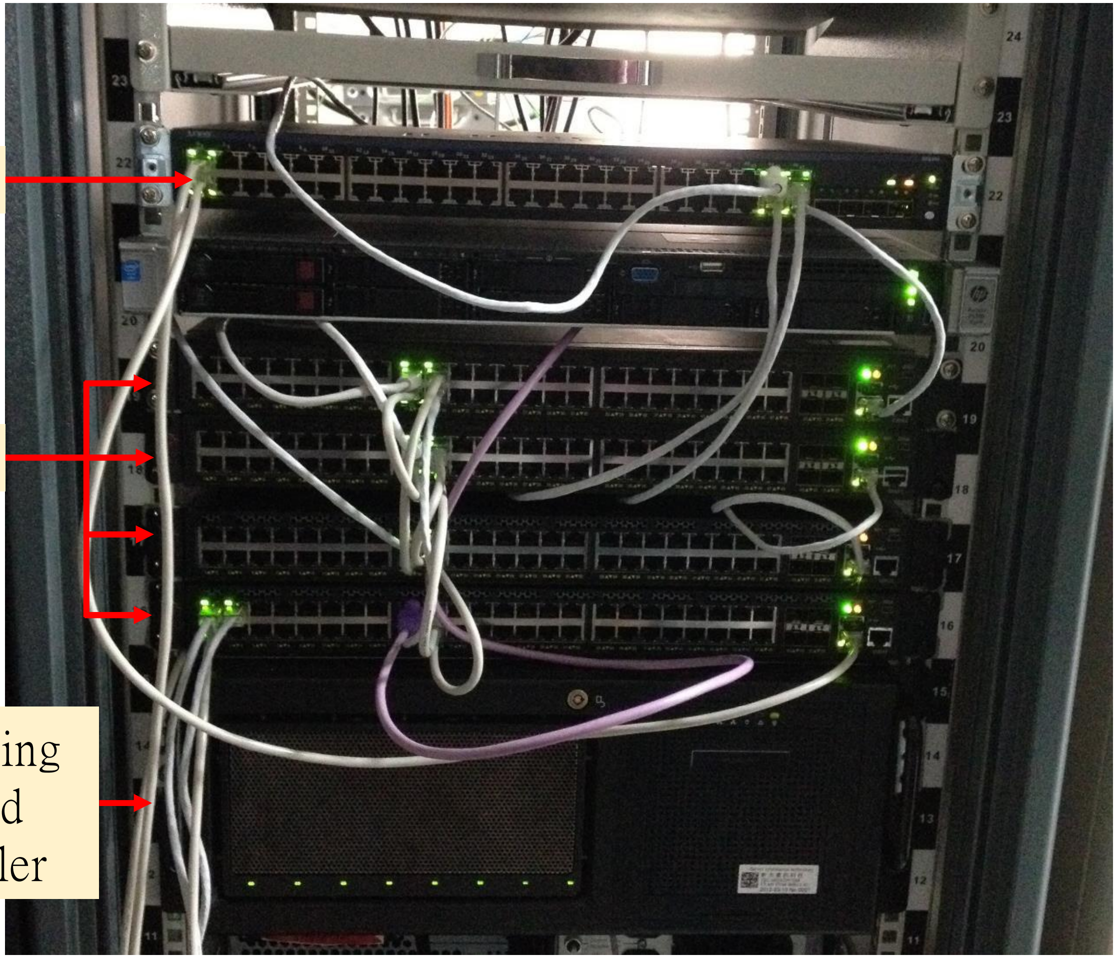
- Motivation
- System Overview
- Problem Formulations & Proposed Solutions
- **Testbed**
- Experiments in Mininet
- Conclusion

Testbed Setup



- 4 OpenFlow switches – Pica8 P3297
- 1 OpenFlow controller - Ryu
- 1 video source - VLS
- 3 video sinks - VLC
- 2 video streams represent 2 descriptors

Testbed



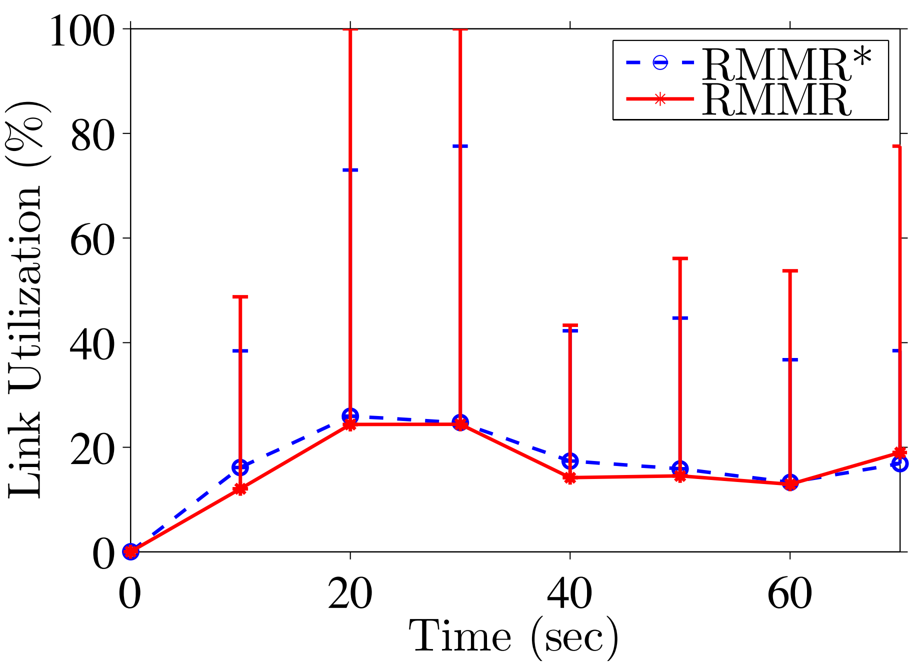
Layer 2 Switch

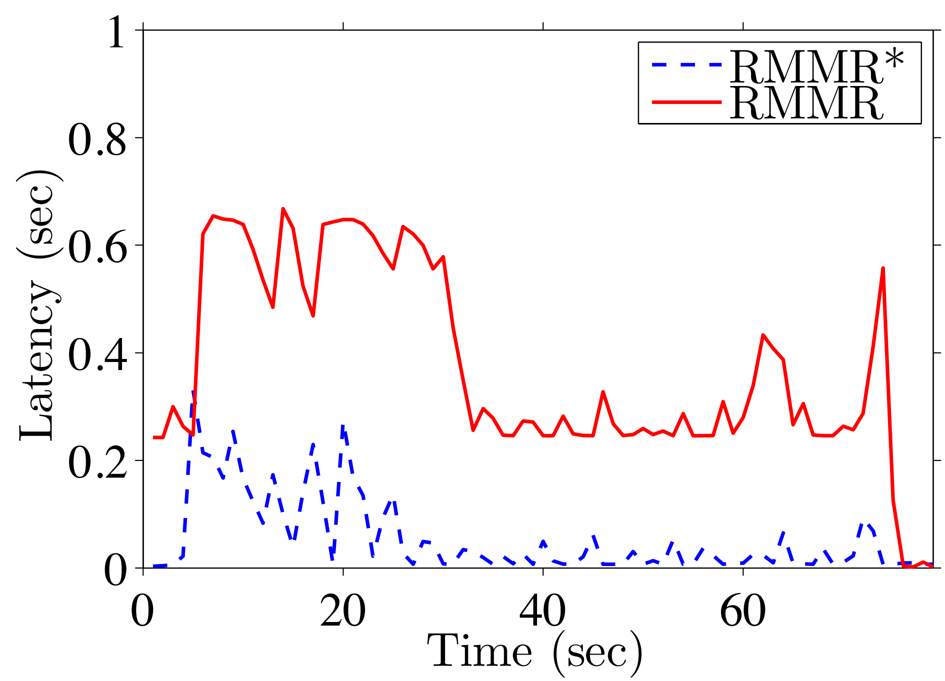
Pica8 P-3297

Server runs streaming server, clients, and OpenFlow controller

Higher Link Utilization Causes Longer Latency

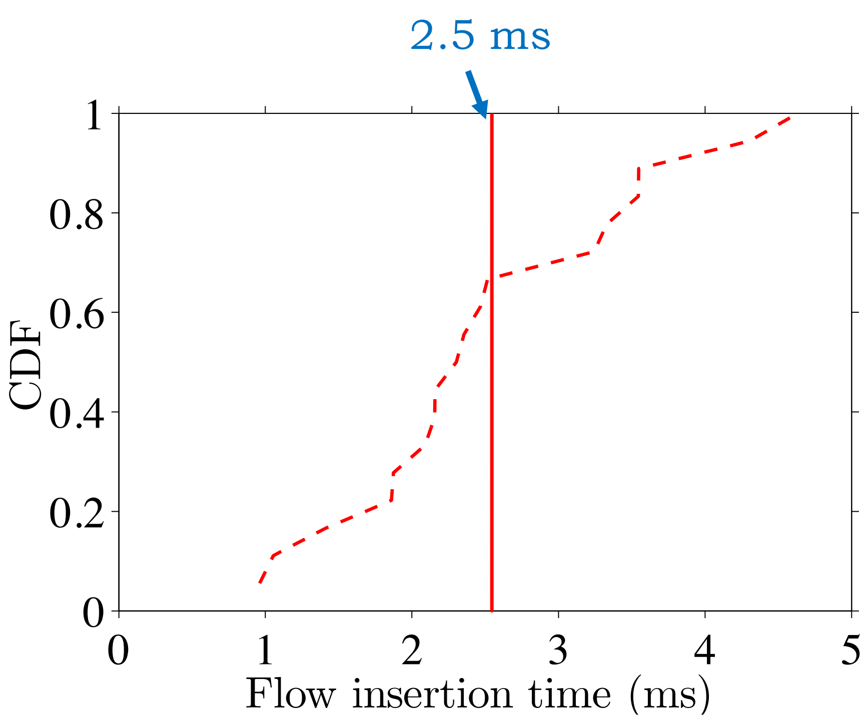
- Link bandwidth: 10 Mbps
- We stream two 4 Mbps videos from source

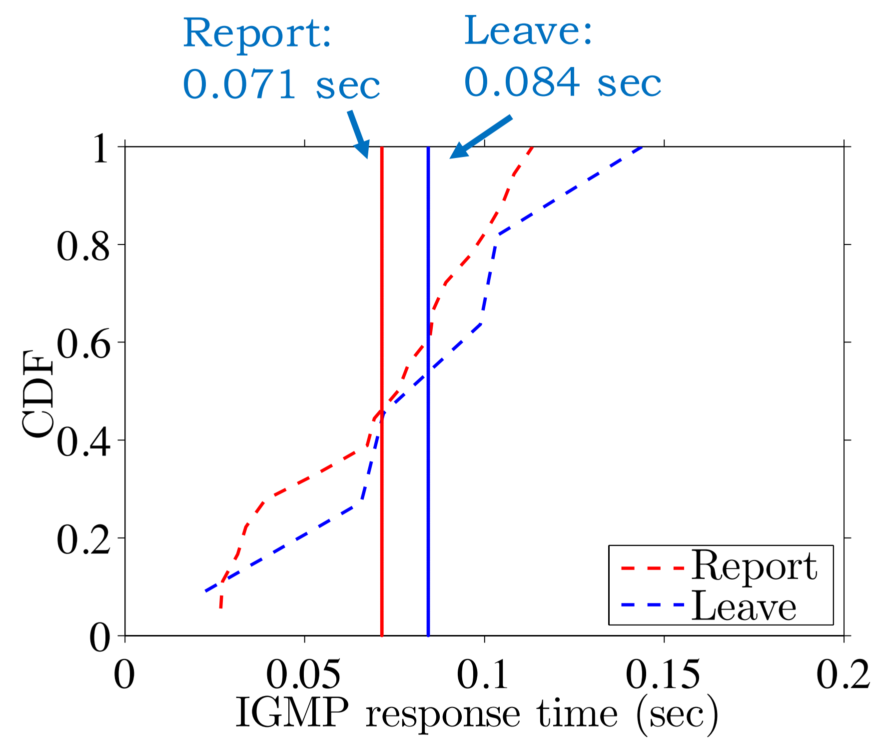




Average packet latency increases about
0.4 sec while link congestion

Our System Incurs Short Response Time





Flow insertion time ranges from **1 to 4.5 ms**

IGMP messages response time ranges from **0.023 to 0.15 sec**

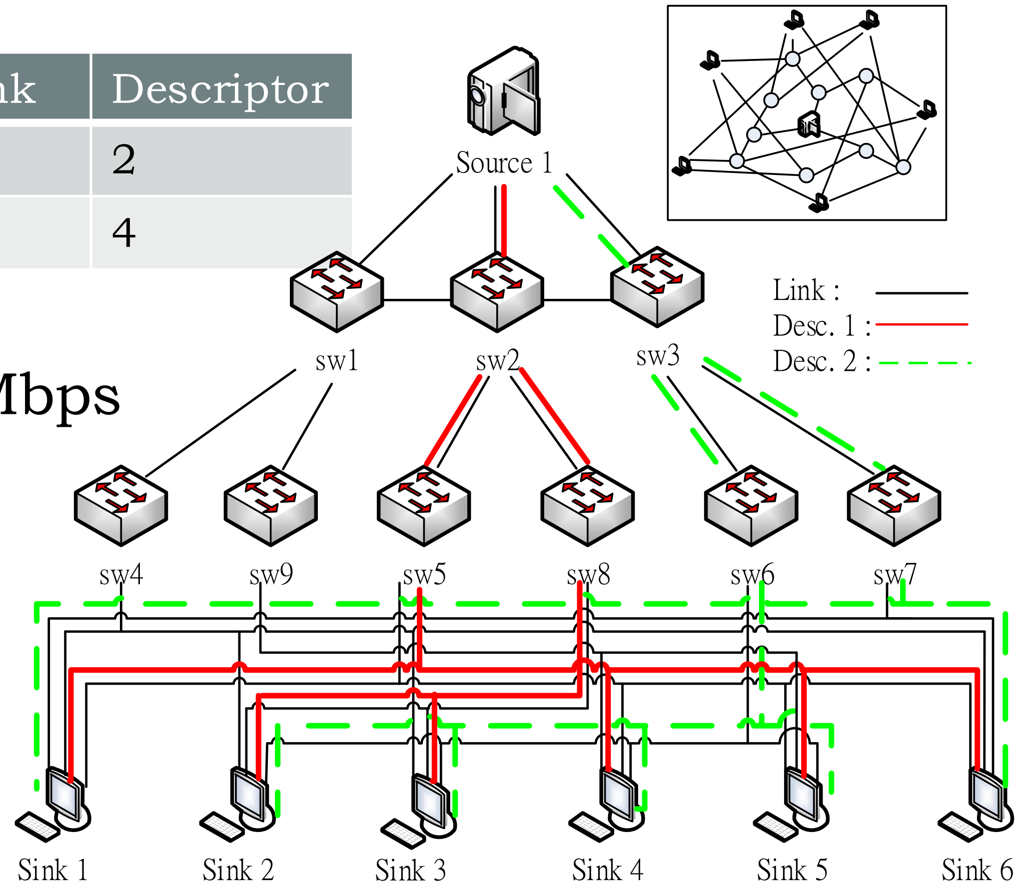
Outline

- Motivation
- System Overview
- Problem Formulations & Proposed Solutions
- Testbed
- **Experiments in Mininet**
- Conclusion

Experiments in Mininet

Topology	Switch	Source	Sink	Descriptor
Small	9	1	6	2
Large	24	4	12	4

- Link bandwidth: 1~5 Mbps
- Algorithm comparison
  - RMMR* - optimal
  - RMMR - heuristic
  - IPM - IP multicast (PIM-SM)

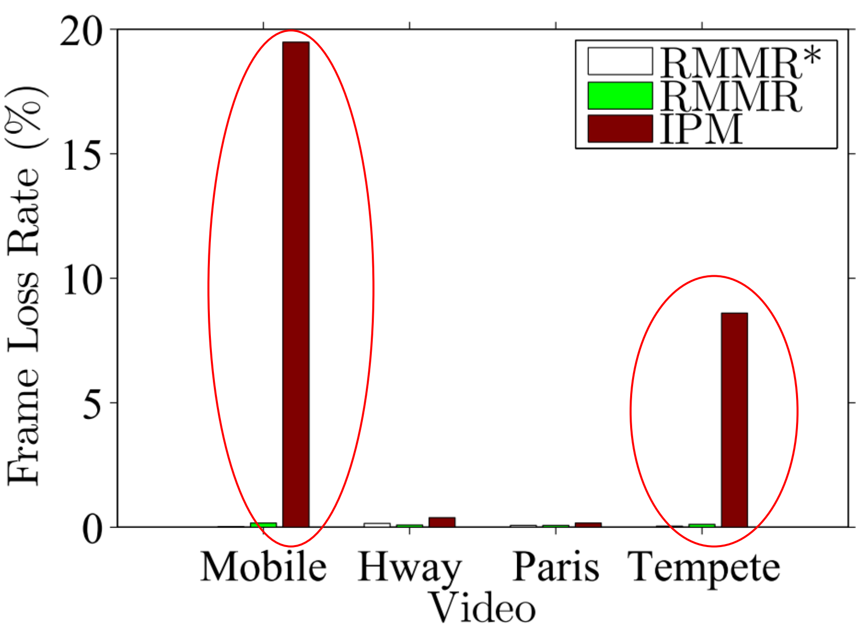


Experiment Setup

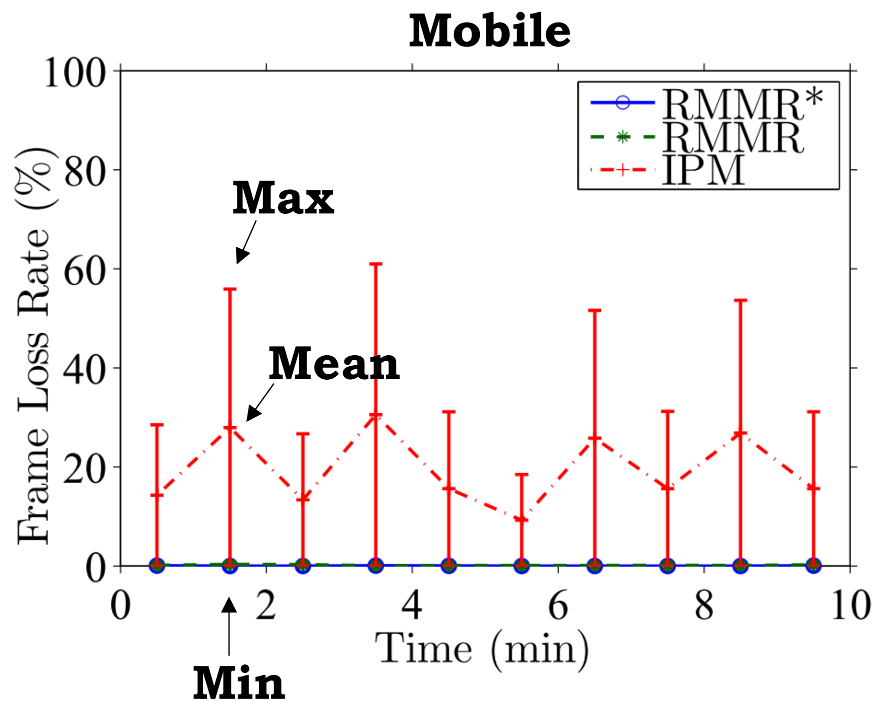
- MDC video traces and their bit rates:
  - Mobile 1.22 Mbps
  - Hway 0.27 Mbps
  - Paris 0.5 Mbps
  - Tempete 0.91 Mbps
- We conduct 10-min experiments with RMMR*,RMMR, and IPM
- The sink join/leave rates = 2 (sinks/minute)

MDC video traces from ASU

Our Algorithms are Bandwidth-aware

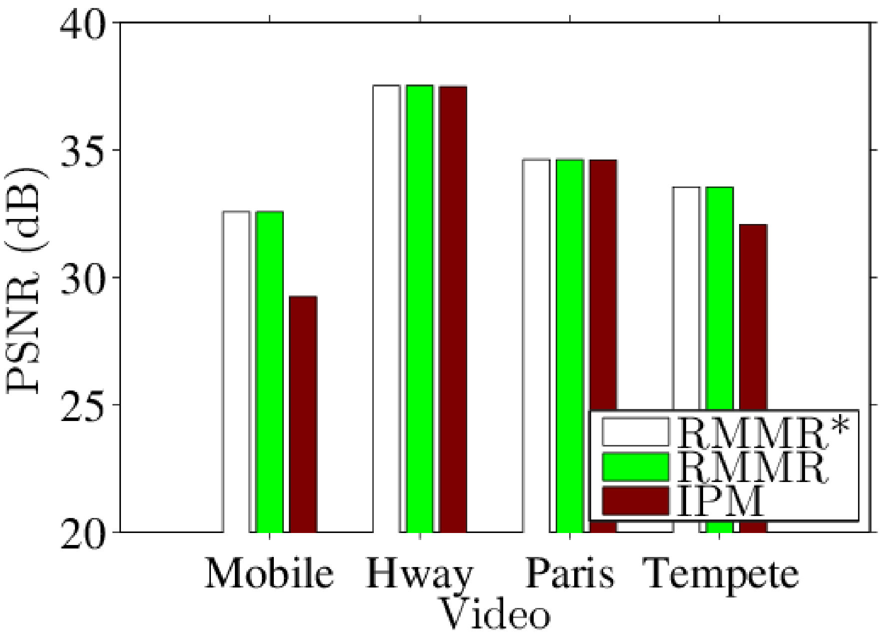


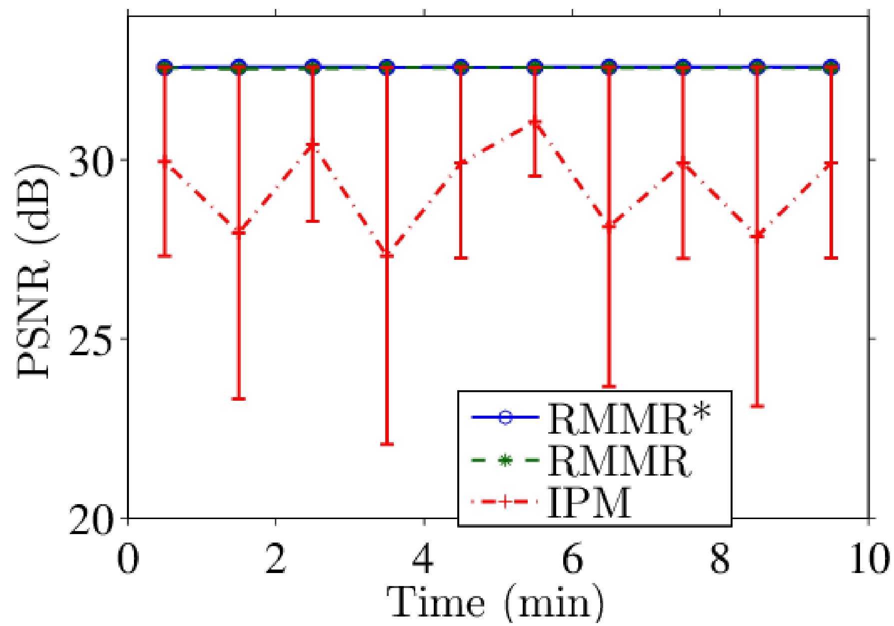
Large bit rate videos get higher frame loss rates using IPM



Our algorithms result in **almost zero frame loss rate**

Our Algorithms Achieve Higher Video Quality

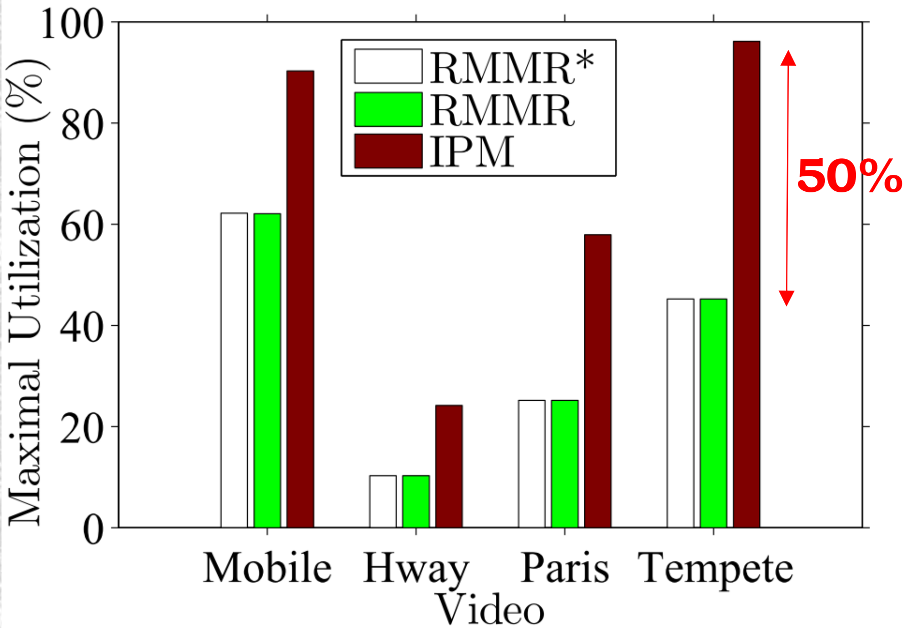


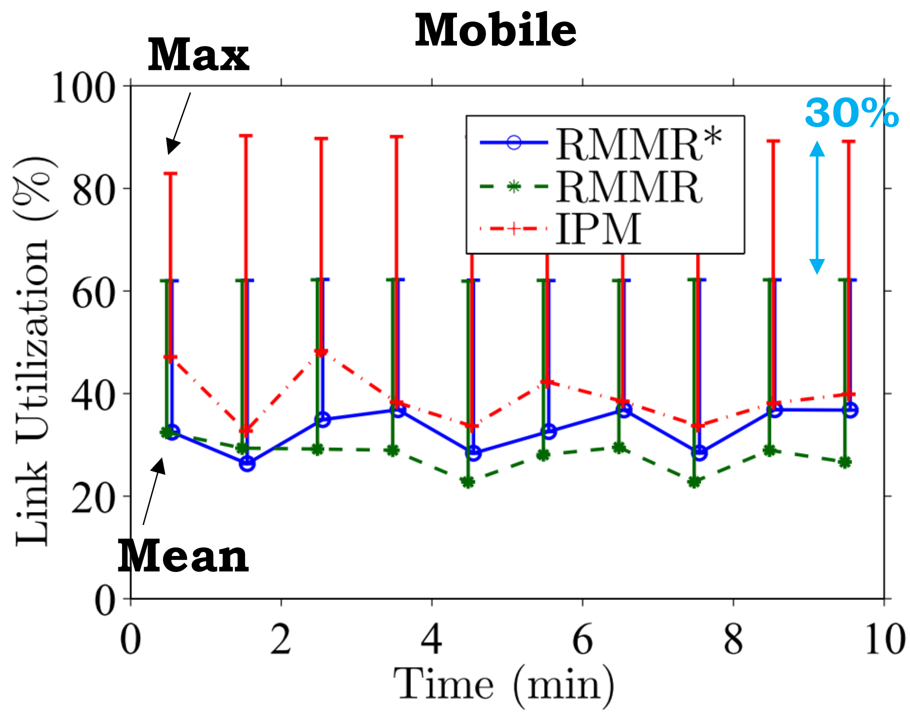


Gaps between RMMR(*) and IPM with **Mobile (4 dB) and Tempete (1 dB)**

IPM leads to at most **10 dB quality fluctuation**

Our Algorithms Reduce Max Link Utilization



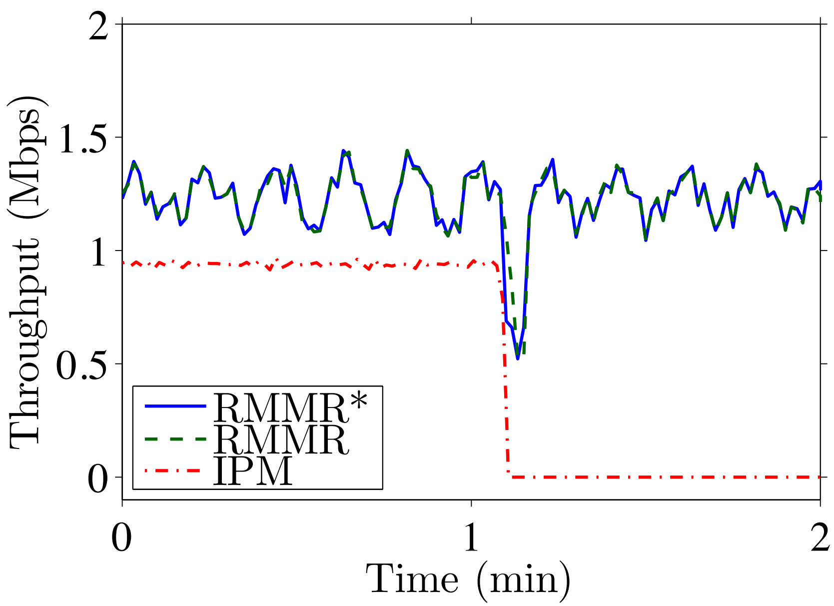


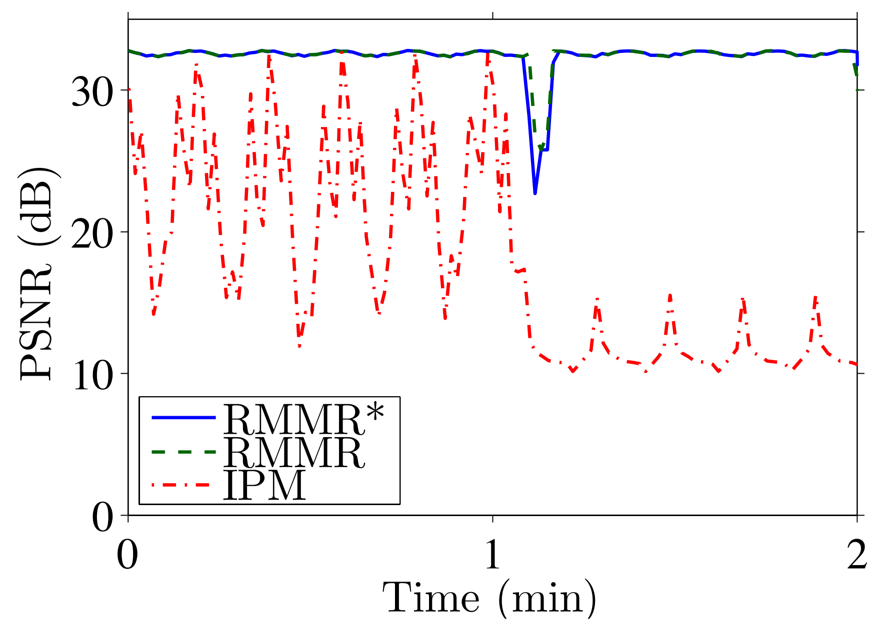
Reduce max link utilization up to 50%

Reduce max link utilization 20~30%

Our Proposed Algorithms are Robust Against Switch Failures

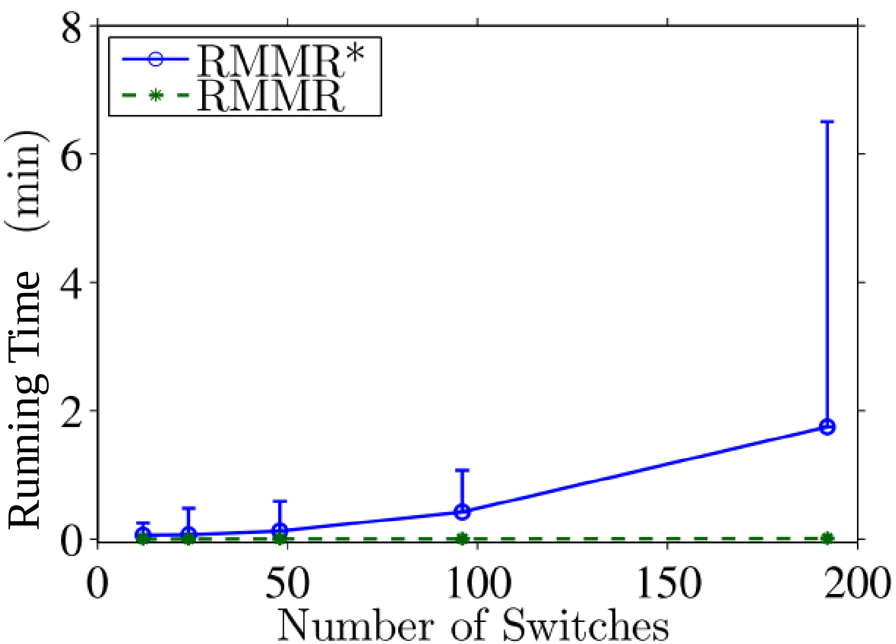
- We shutdown switch 1 at 65 sec

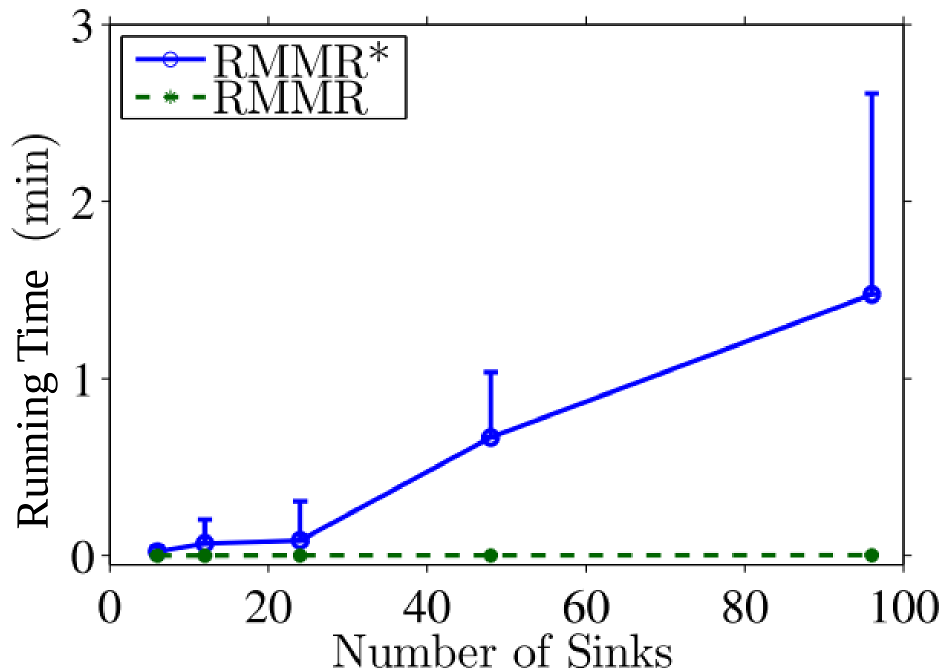




Our system **recovers the routes in few seconds** after switch failure

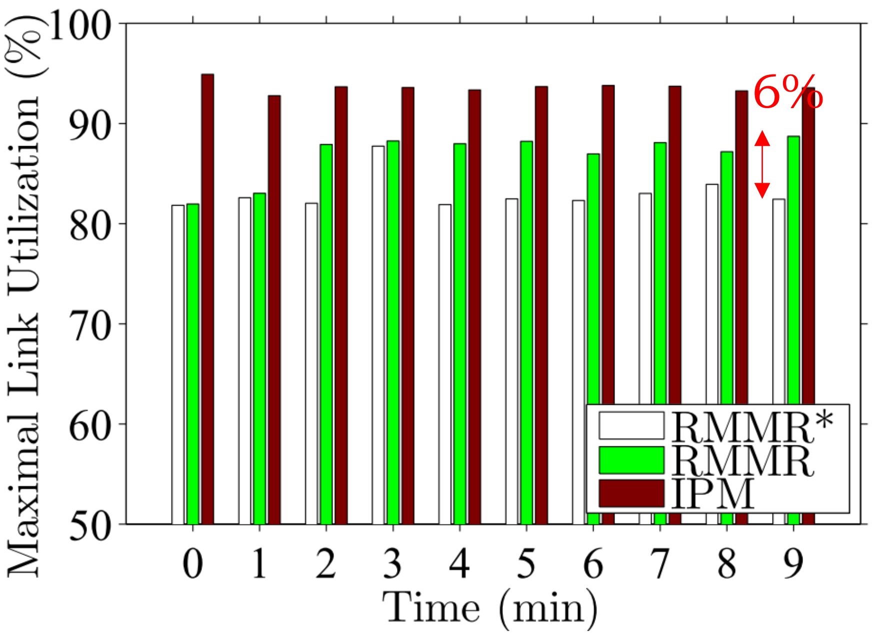
The Scalability of Our Algorithms

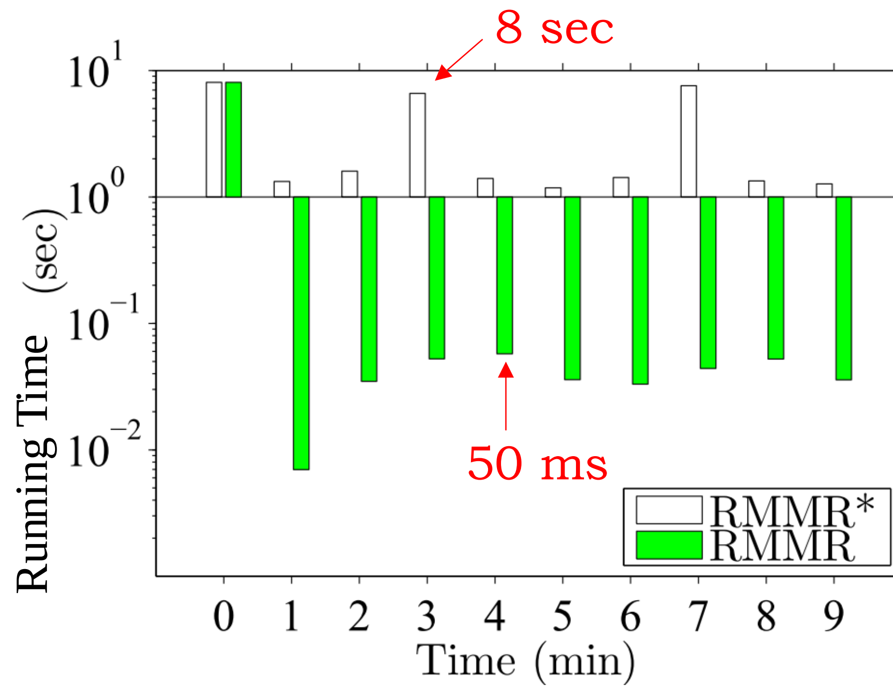




RMMR always terminates in < 0.2 sec

Tradeoff Between Optimality and Running Time





RMMR* has higher optimality: Reduce max link utilization up to 6%

RMMR is more efficient: Reduce running time from 8 sec to 50 ms

Outline

- Motivation
- System Overview
- Problem Formulations & Proposed Solutions
- Testbed
- Experiments in Mininet
- **Conclusion**

Conclusion

- Multicast on SDNs
  - Reduce setup cost and maintenance cost
  - Use a global view to get an optimal solution for multicast routing
- RMMR$^{(*)}$ algorithm
  - Robust, load balanced, and flexible
  - **RMMR*:** Optimal solution, suitable for smaller and more static networks
  - **RMMR**: Efficient solution, suitable for larger and more dynamic networks

Future Work

1. Consider background traffic to compute the multicast routing depends on the current traffic
2. Design an adaptive algorithm, which automatically makes decisions on when to update the multicast routing with optimal solution

- Thank you !