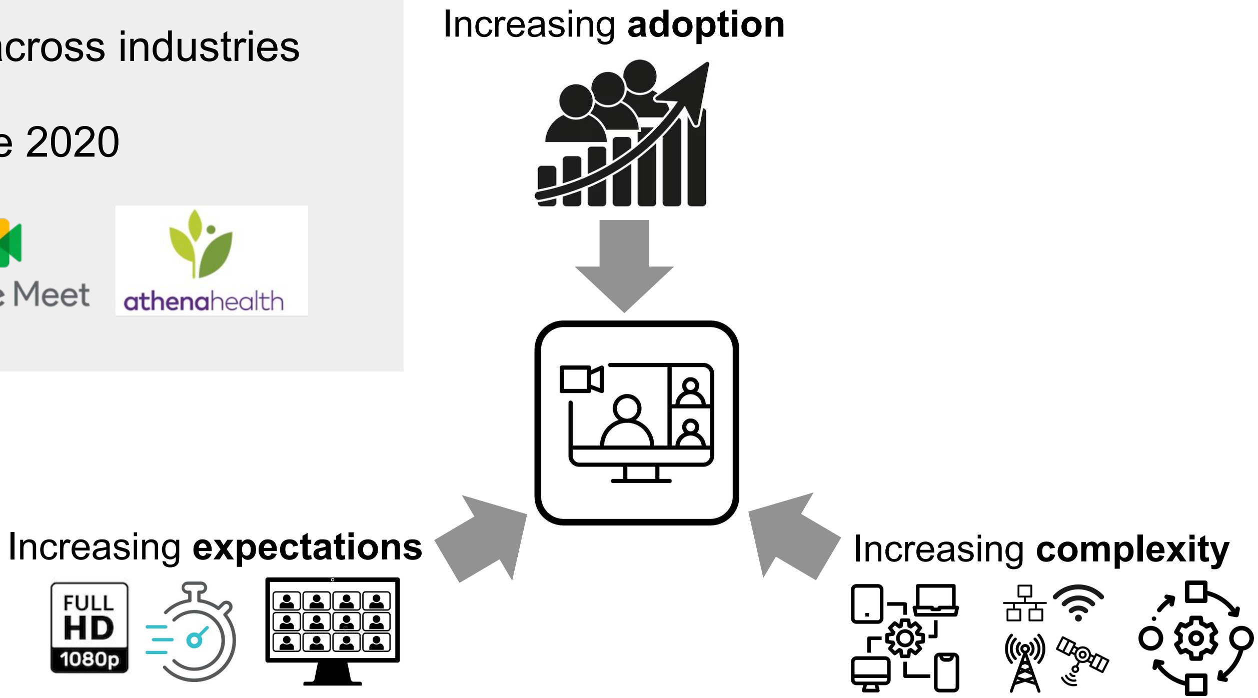
Scallop: Scalable Video Conferencing Using SDN Principles

Oliver Michel, Satadal Sengupta, Hyojoon Kim,
Ravi Netravali, Jennifer Rexford

ILLINOIS TECH PRINCETON UNIVERSITY UNIVERSITY of VIRGINIA

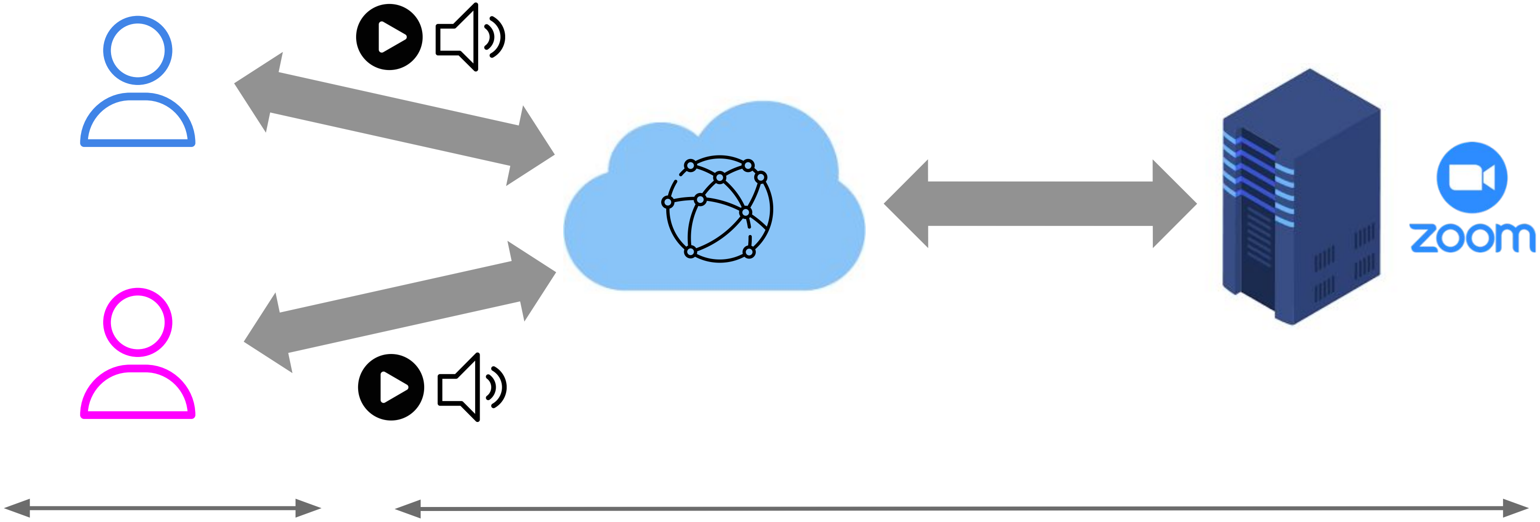
The Future of Video Conferencing

- Essential application across industries
- Explosive growth since 2020



Challenge: Provide consistently high quality at scale

The Missing Middle

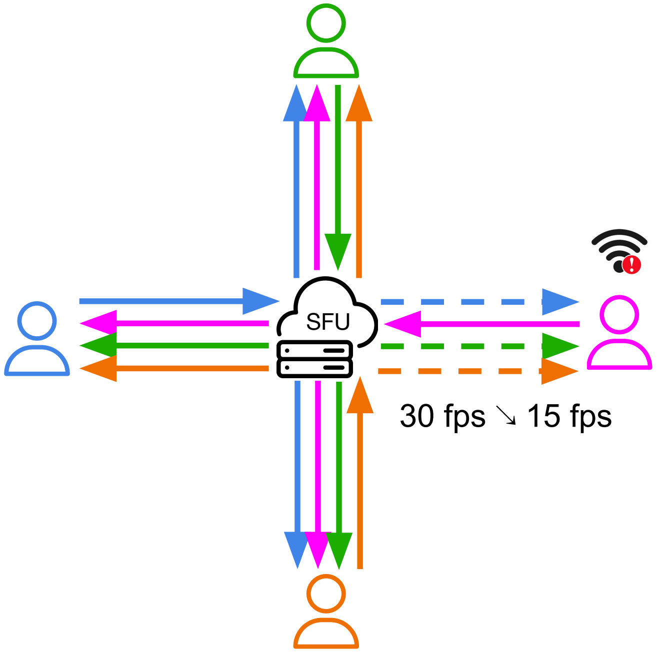


e.g., Video Codecs,
Rate-Adaptation Algorithms,
Measurement Studies

?

In this study: The application operators' perspective

Selective Forwarding Unit (SFU)



SFU roles:

(1) Relay audio and video streams

(2) Monitor and adapt media signals

SFUs hard to scale:

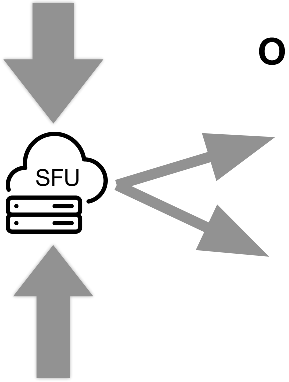
(1) Workload hard to predict

(2) Quadratic scaling

3 → 4 parts. ⇒ 9 → 16 streams

The SFU Scaling Challenge

Dynamic, high-volume workload

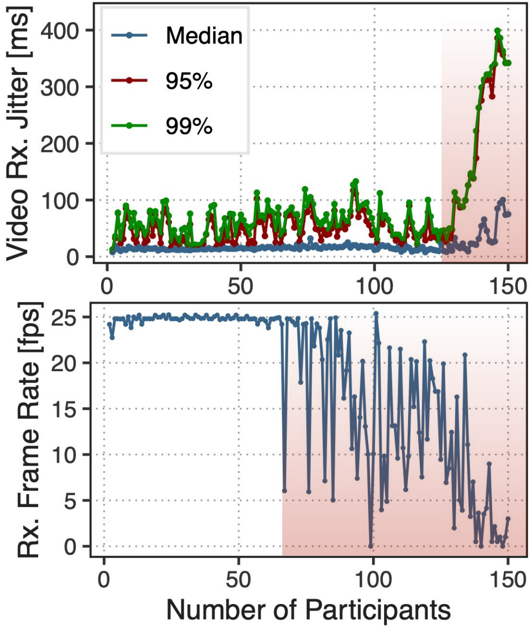


Underprovisioning can affect quality massively

Operators left with two options:

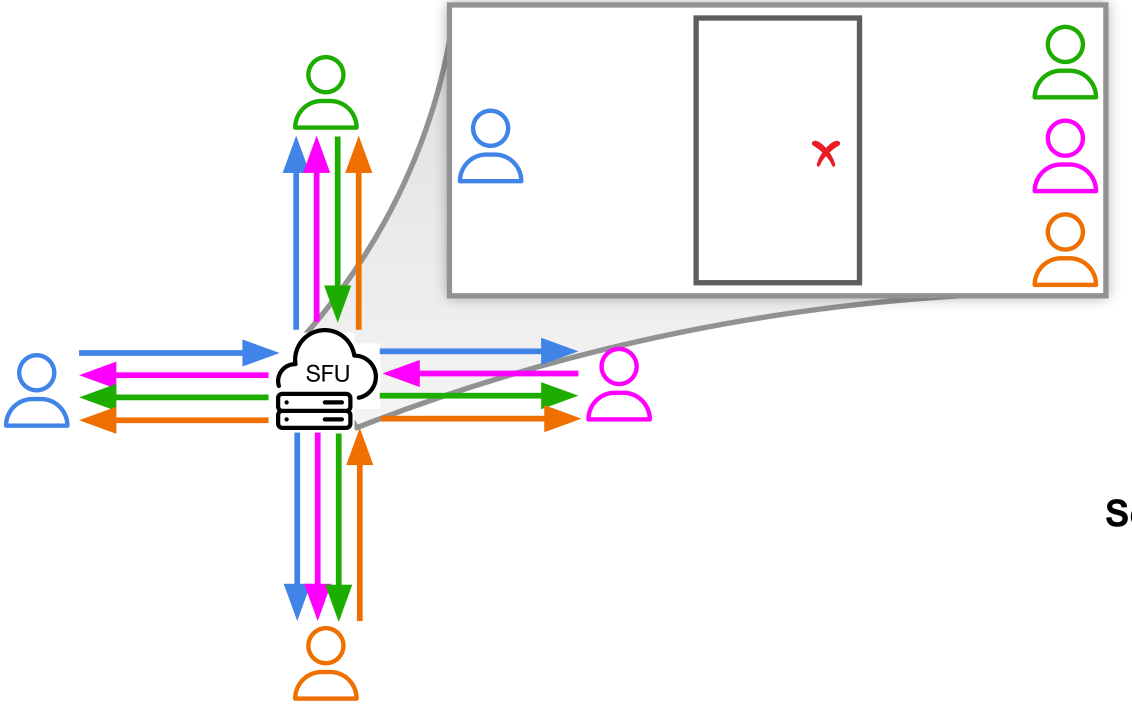
- Massively over-provision
  → costly and wasteful
- Reactively autoscale
  → risk harming QoE for users

QoE in MediaSoup when increasing SFU load:



SFUs as Packet Processors

💡 SFU operation is strikingly similar to traditional packet processing



(1) Relay audio and video streams

↓

Replicate traffic *(Multicast)*

(2) Monitor and adapt media signals

↓

Selectively forward traffic *(Firewall)*

Scallop

Fundamental rethink of video conferencing infrastructure
to support long-term traffic forecasts

A novel hardware/software SFU co-design inspired by SDN

SFU

Efficient data plane → relays high-volume media streams using line-rate hardware

Software control plane → handles critical but infrequent tasks

- Offload >99% traffic to hardware
  - up to 128K concurrent meetings
  - ~ 10–200× scale over software
  - at comparable cost
  - reduce latency by 27×

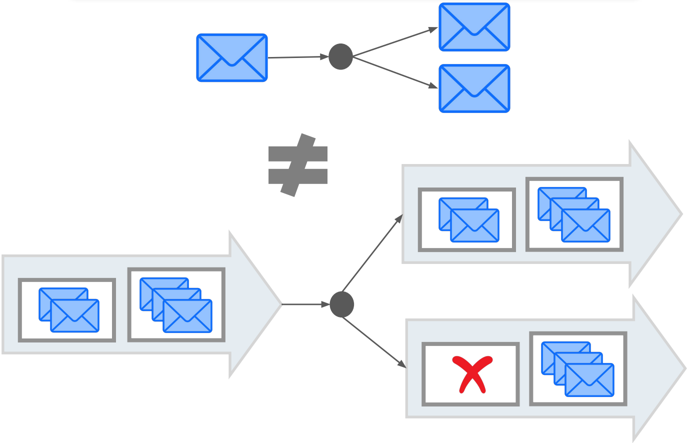
Challenges

Monolithic Software Architecture

1

How to disaggregate into control and data planes?

Complex Multicast Semantics

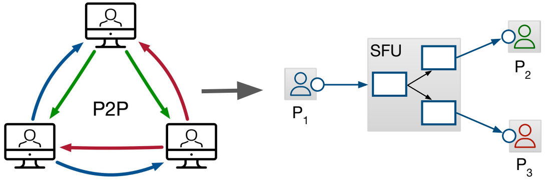


2

How to realize and scale application-layer multicast?

Misaligned with Widely-Deployed Standard

WebRTC



3

How to make Scallop interoperable with WebRTC?

1 SDN-Inspired Disaggregation

SFU workload is amenable to a control/data-plane split

(1) >10ms latency, every few mins.

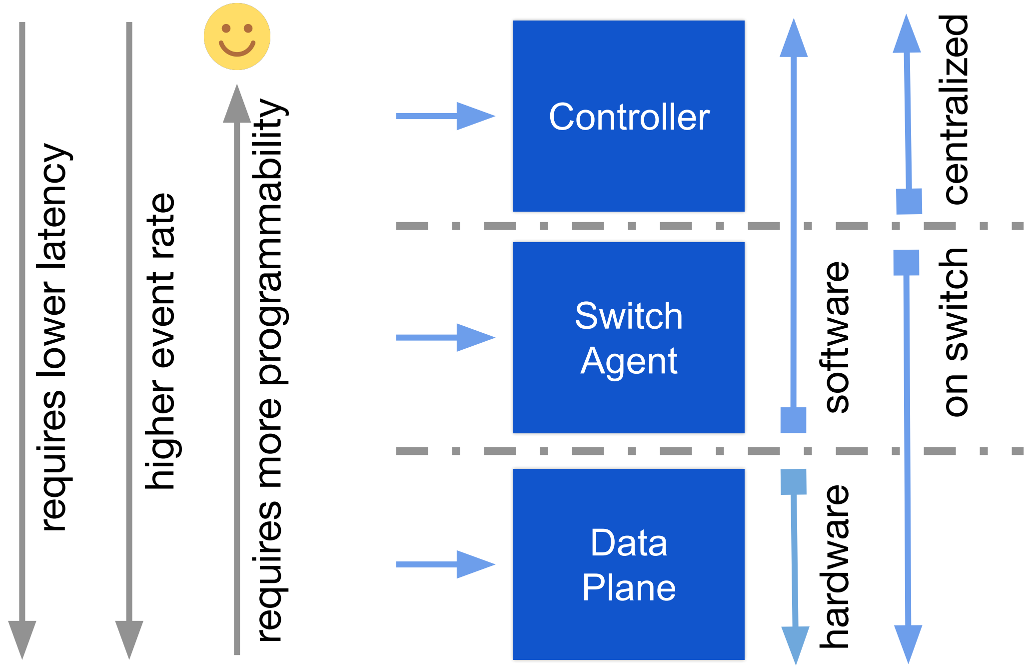
e.g., session management, signaling

(2) 1 < t < 10ms latency, 2-3 per sec.

e.g., handling feedback messages and connectivity checks

(3) <1ms latency, 100s per sec.

replication and selective forwarding of media packets

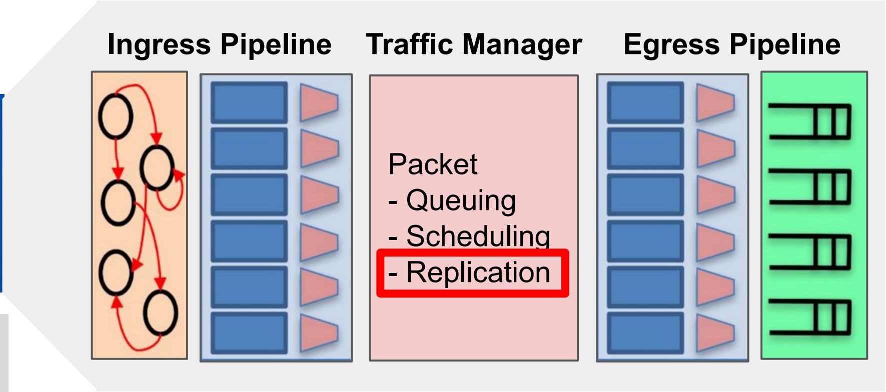


2 Scalable Application-Layer Multicast

Hardware-native packet copying capabilities can be leveraged for SFU-style replication

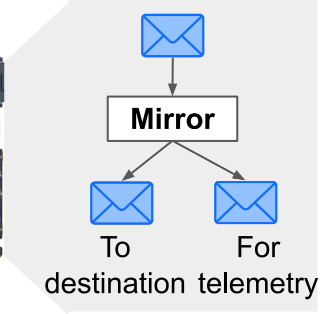
Programmable Switches

Intel Tofino2



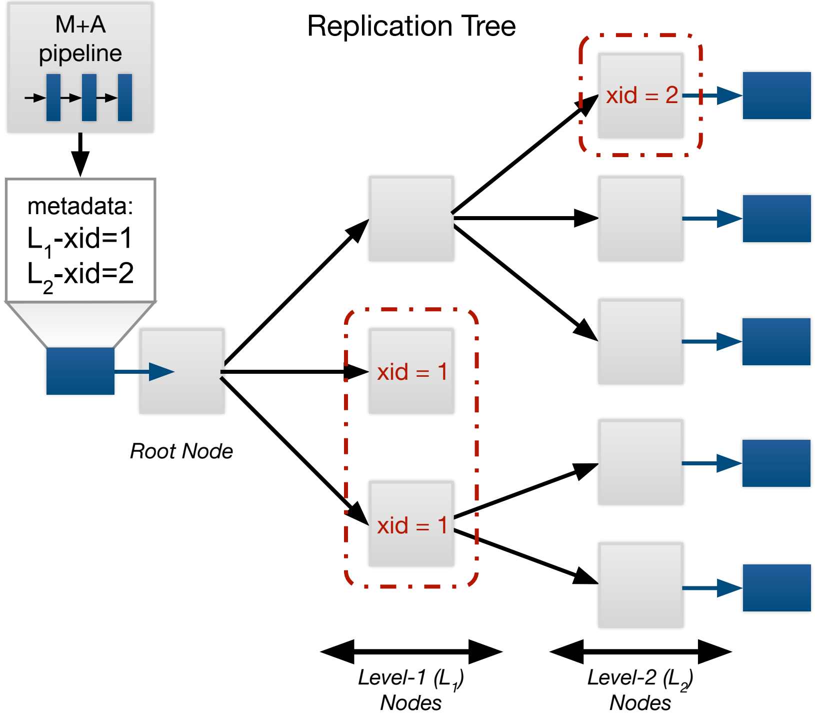
SmartNICs/DPUs

NVIDIA BlueField-3



2 Scalable Application-Layer Multicast

Background on Tofino's Packet Replication Engine (PRE)



- Abstraction: *replication tree* with *level-1 nodes* and *level-2 nodes*
- Supports *dynamic tree pruning*
  - Each node can be associated with an exclusion ID (xid)
  - The ingress match+action pipeline can associate individual packets with xids
  - No replication along edges leading to excluded nodes

2 Scalable Application-Layer Multicast

Packet Replication in Scallop: Challenges

PRE-to-VC Mapping?

- PRE entities:
  - Root, L1/L2 nodes
  - L1/L2 xids
- VC entities:
  - Meetings, participants
  - Quality layers

Different Meeting Configurations

- Rate adaptation status
- Two-party vs. multiparty
- Can change dynamically

Limited Resources

- 64,000 replication trees
- 2^{24} L1 nodes

Q

How to (i) correctly and (ii) efficiently map VC entities to PRE entities for each meeting configuration?

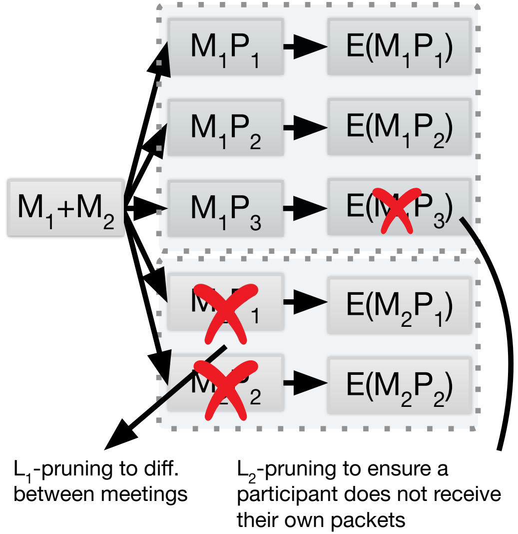
2 Scalable Application-Layer Multicast

Packet Replication in Scallop: Solution

Optimal Designs

- Non-Rate-Adapted (NRA)
- Rate-Adapted (RA)
  - Receiver (RA-R)
  - Sender-Receiver (RA-SR)
- Two-Party (2P)

NRA Design



Supports upto **128K concurrent meetings** and **2^{24} concurrent participants**

Evaluation

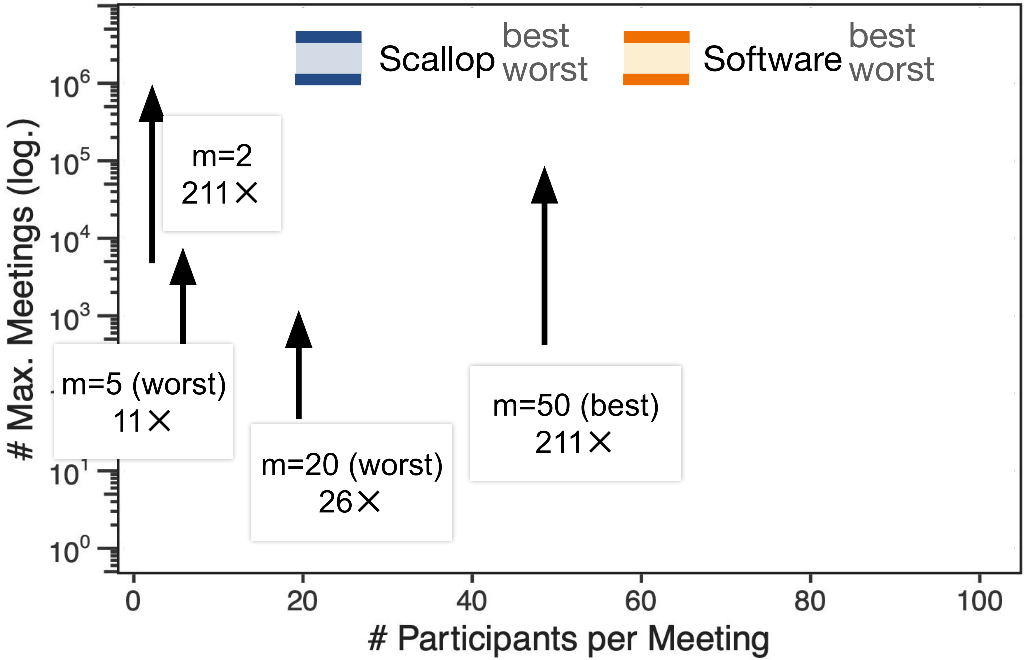
Scallop processes >99% traffic in hardware

Protocol/Type	Packets	%	Per sec.	KBytes	%
RTP	170,870	94.5	284.3	166,762	99.47
- Audio	29,746	16.46	49.49	3826	2.28
- Video	141,124	78.09	234.81	162,935	97.19
- AV1 DS*	5	«	0.008	6	«
RTCP	9,153	5.06	15.22	801	0.48
- SR/SDES	3,456	1.91	5.75	304	0.18
- RR*	240	0.39	0.13	15	0.01
- RR/REMB*	5,457	3.02	9.07	482	0.29
STUN*	**695**	**0.38**	**1.15**	89	0.05
Control Plane	6,397	3.54	10.64	593	0.35
Data Plane	174,326	96.46	290.06	167,066	99.65
Total	180,718	100	300.69	167,653	100

✓ Scallop's control/data-plane split is effective in reducing software load

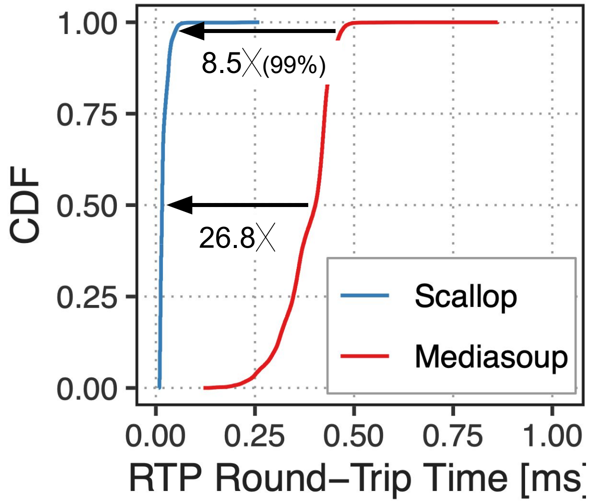
Evaluation

Scallop improves scalability over software by 7× to 211×



Evaluation

Scallop reduces SFU-induced latency: median by 27× and tail by 8×



Conclusion

- **Scallop:** Novel, hardware-software co-designed SFU:
  - >99% of traffic in hardware
  - 7 – 211✕ scale improvement over software at comparable cost
  - Reduce SFU-induced latency by 27✕

- **Artifacts on GitHub:**
  - Control plane + software model of data plane
  - Hardware prototypes:
    - Intel Tofino2 switch
    - NVIDIA BlueField-3 DPU
  - Wireshark plugin

Thank You!

- Code: https://github.com/Princeton-Cabernet/Scallop
- Contact: satadal.sengupta@princeton.edu

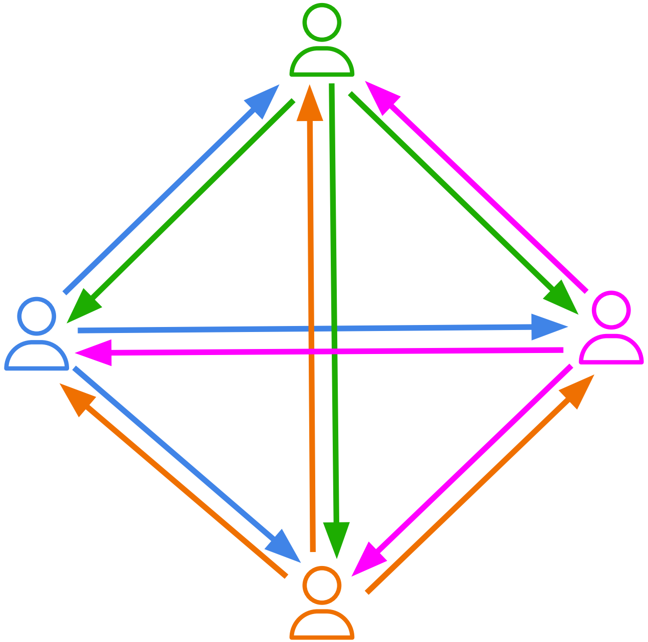
I'm on the market for faculty positions
in the US, Canada, and Europe

ILLINOIS TECH PRINCETON UNIVERSITY UNIVERSITY of VIRGINIA

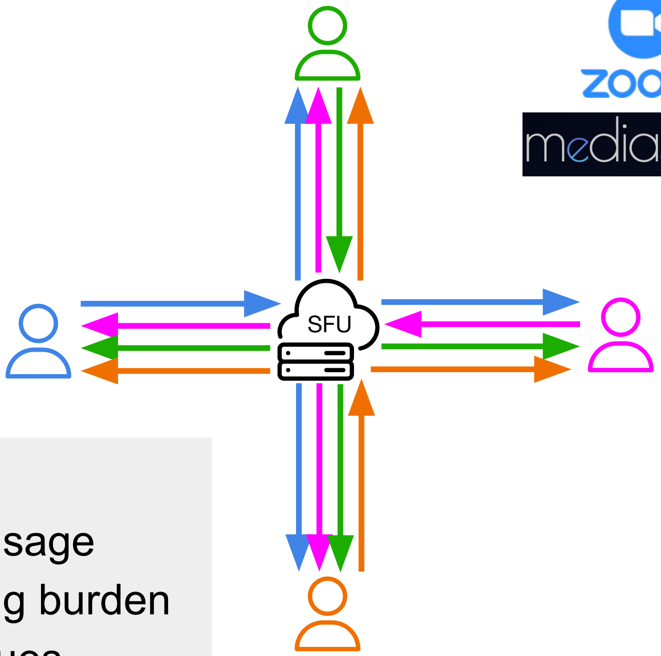
Backup

Video-Conferencing Infrastructure

Peer-to-Peer Architecture



Selective Forwarding Unit (SFU)



zoom

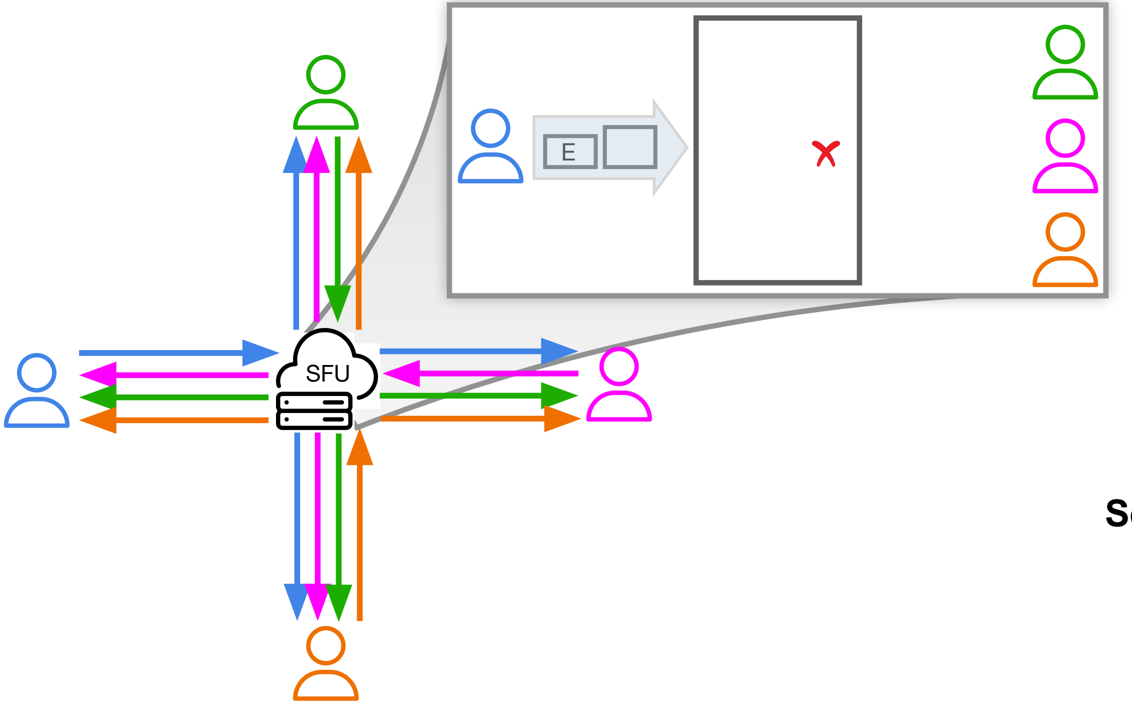
mediasoup

Decreases:

- Uplink usage
- Encoding burden
- NAT issues

SFUs as Packet Processors

SFU operation is strikingly similar to traditional packet processing



(1) Relay audio and video streams

Replicate traffic *(Multicast)*

(2) Monitor and adapt media signals

Selectively forward traffic *(Firewall)*

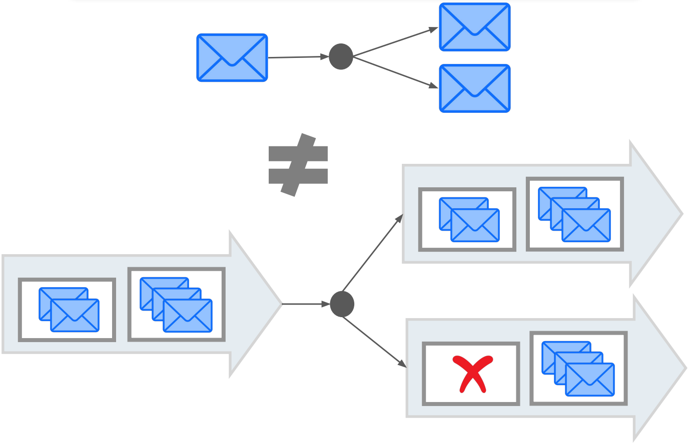
Challenges

Monolithic Software Architecture

1

How to disaggregate into control and data planes?

Complex Multicast Semantics

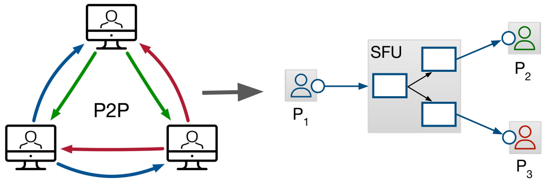


2

How to realize and scale application-layer multicast?

Misaligned with Widely-Deployed Standard

WebRTC

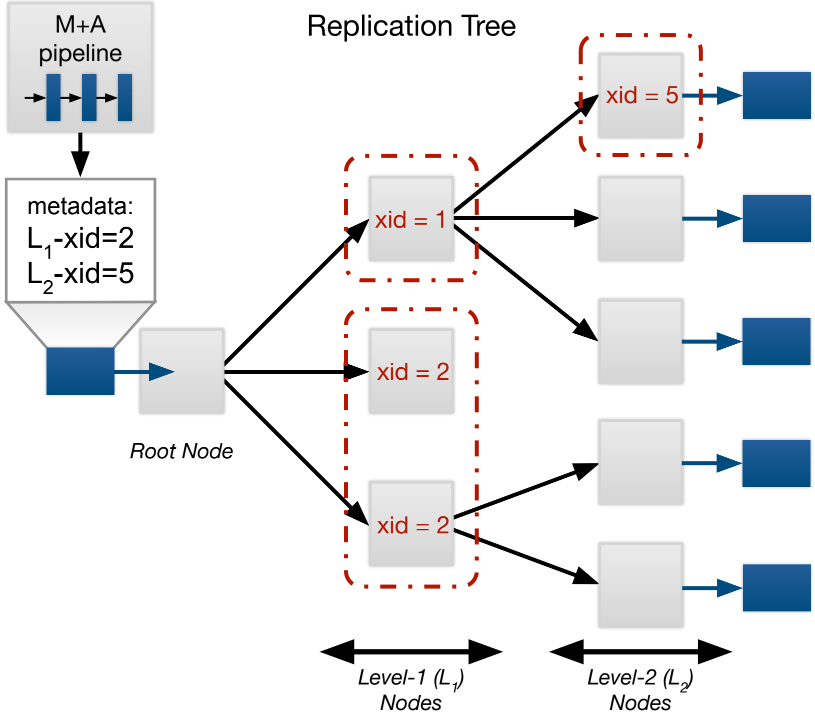


3

How to make Scallop interoperable with WebRTC?

2 Scalable Application-Layer Multicast

Background on Tofino's Packet Replication Engine (PRE)

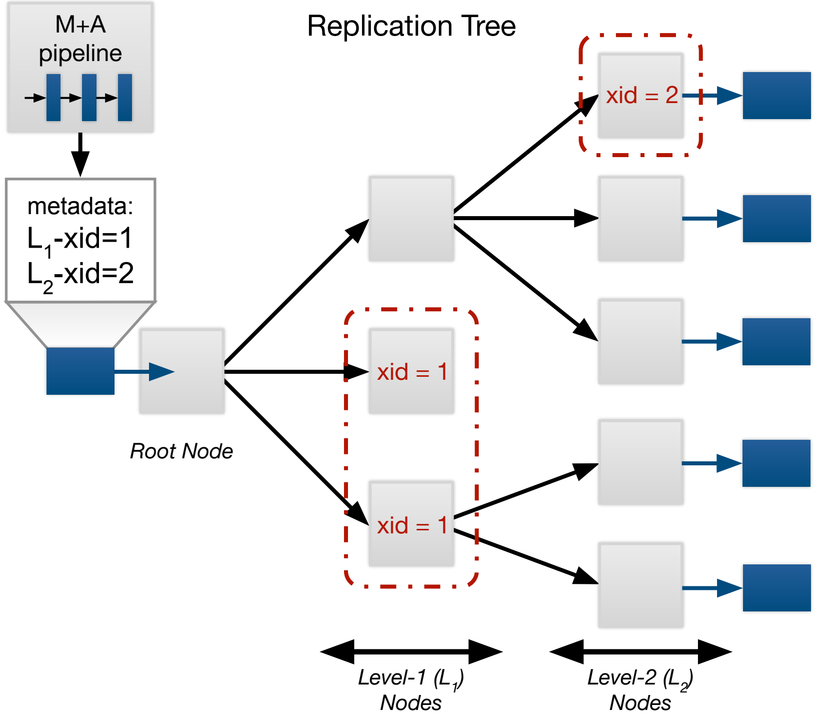


- Abstraction: *replication tree* with *level-1 nodes* and *level-2 nodes*

- Supports *dynamic tree pruning*
  - Each node can be associated with an exclusion ID (xid)
  - The ingress match+action pipeline can associate individual packets with xids
  - No replication along edges leading to excluded nodes

2 Scalable Application-Layer Multicast

Background on Tofino's Packet Replication Engine (PRE)



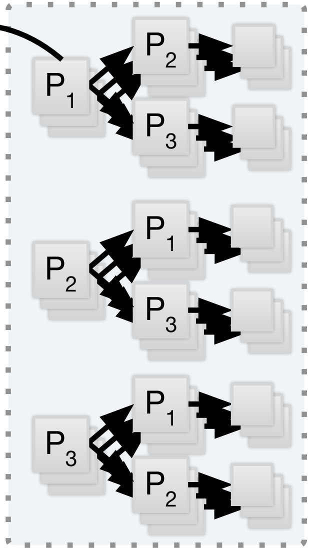
- Abstraction: *replication tree* with *level-1 nodes* and *level-2 nodes*
- Supports *dynamic tree pruning*
  - Each node can be associated with an exclusion ID (xid)
  - The ingress match+action pipeline can associate individual packets with xids
  - No replication along edges leading to excluded nodes

2 Scalable Semantics-Aware Replication

Packet Replication in Scallop

Straw Man

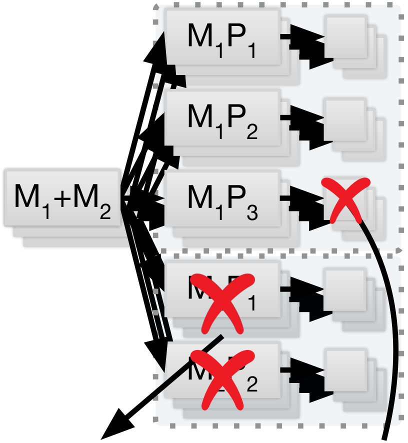
one tree per quality layer



$N = \tau / \kappa = 64{,}000/3 \approx 21{,}333$

$M \leq N$

Optimization



L_1-pruning to diff. between meetings (max. 2 per tree)

L_2-pruning to ensure a participant does not receive their own packets

$N \leq \upsilon$

$M = 2\tau / \kappa = 2 \times 64{,}000/3 \approx 42{,}666$

Other Designs

- Non-Rate-Adapted (NRA)
- Rate-adapted/Receiver (RA-R)
- Rate-adapted/Sender-Receiver (RA-SR)
- 2-Party (2P)

Intel Tofino2:
- 64,000 replication trees (τ)
- 2^{24} L_1 nodes (υ)

Scallop Prototype:
- 3 quality layers (κ)

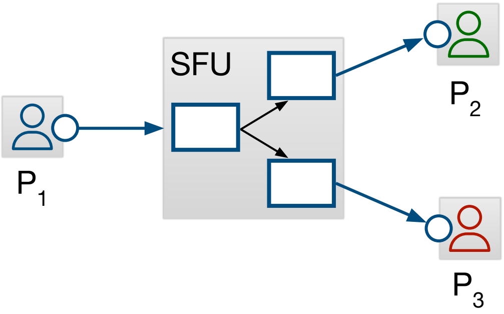
↓

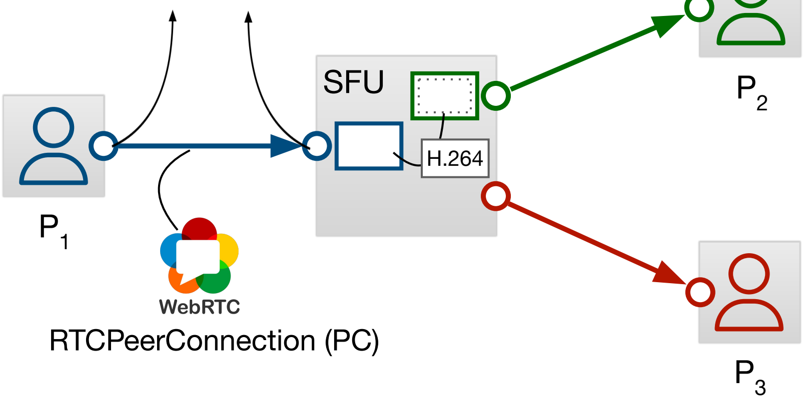
Maximize:
- N # of participants
- M # of meetings

3 Interoperability with WebRTC

PC Endpoints

- congestion control
- encryption/decryption
- packetization/de-packetization





Proxy SFU Architecture

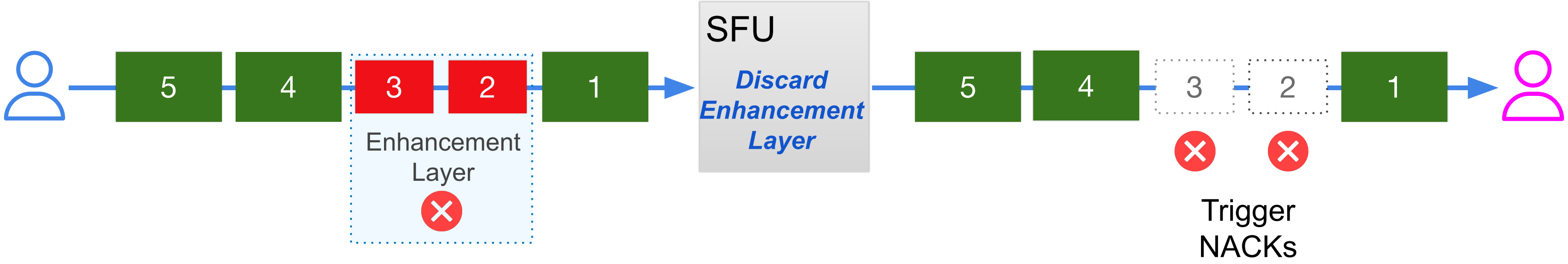
- hardware-friendly
- low overhead at SFU
- latency-friendly

Split-Proxy SFU Architecture

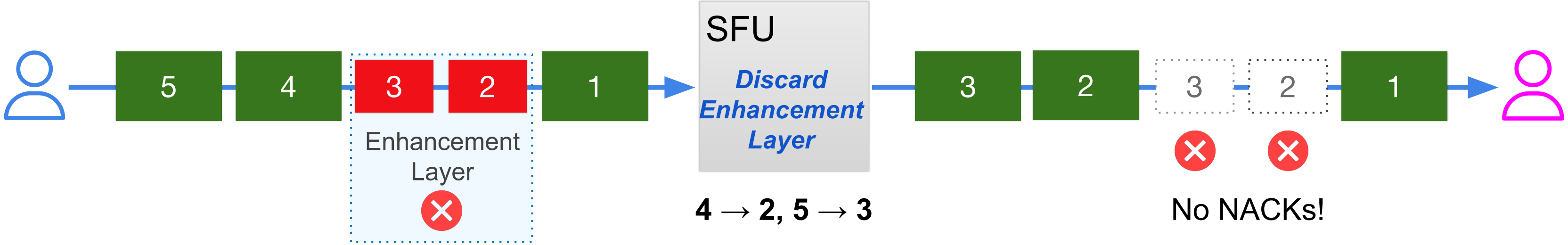
- difficult in hardware
- lots of replicated logic at SFU
- introduces latency

3 Interoperability with WebRTC

Major downstream challenge: Transparent rate adaptation in the data plane

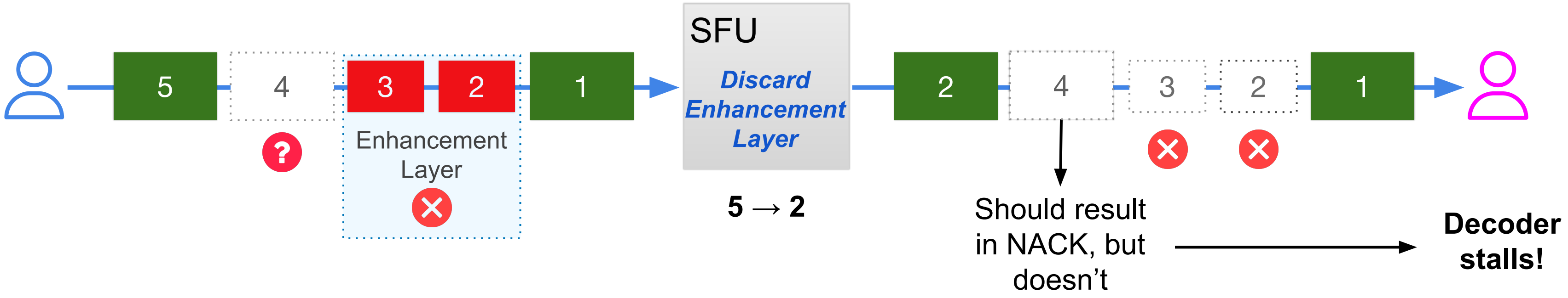


Solution: Rewrite sequence numbers at the SFU with an *offset*



3 Interoperability with WebRTC

Challenge: Naive rewriting causes video freeze during network loss



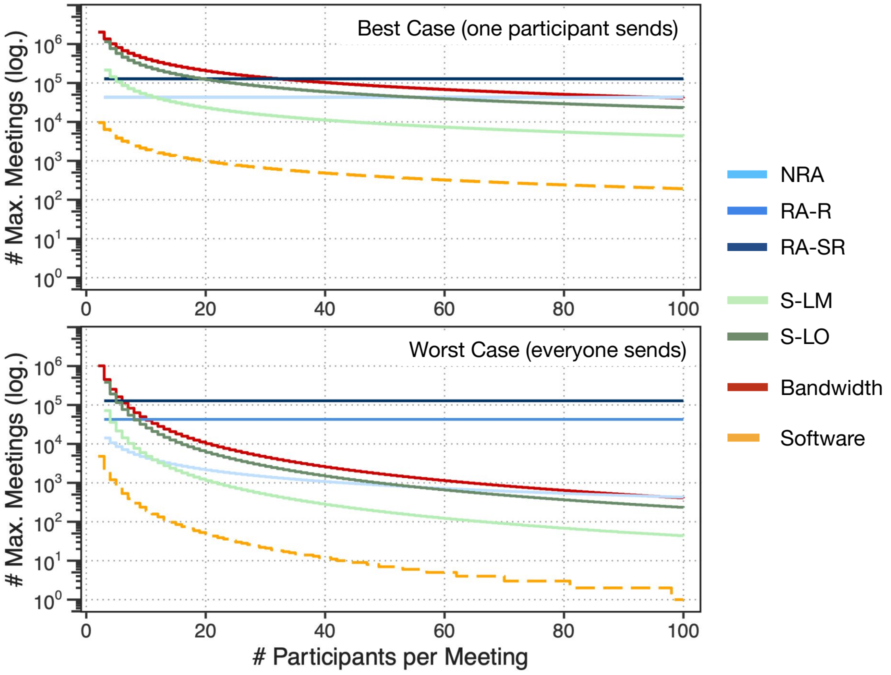
Observation: When unsure, leaving gap better than hiding one

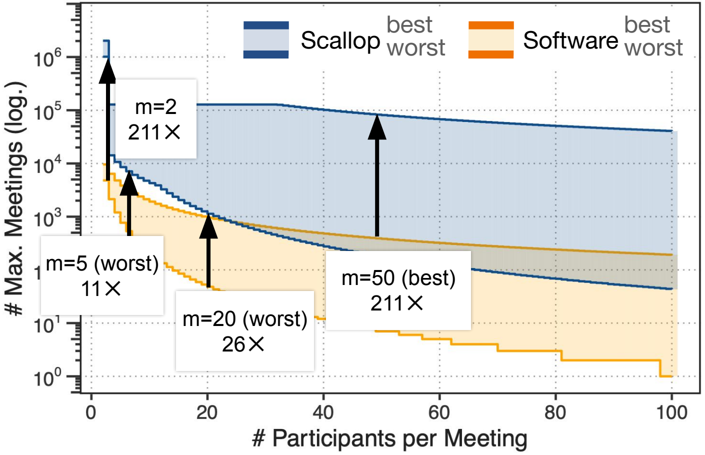
Hardware-friendly heuristic that never hides loss at cost of possibly unnecessary retxs.

Two variations based on trade-off between unnecessary retxs.
and switch memory (S-LO, S-LM)

Evaluation

Scallop improves scalability over software by 7× to 211×





- Scale improvement depends on meeting composition and rate-adaptation characteristics
- Scallop improves scalability 7× to 211× over software and always performs better than software *(for a given meeting composition)*