

# XCo: Explicit Coordination for Preventing Congestion in Data Center Ethernet

Vijay Shankar Rajanna, Smit Shah, Anand  
Jahagirdar and Kartik Gopalan

Computer Science, State University of New York  
at Binghamton

Email: [kartik@binghamton.edu](mailto:kartik@binghamton.edu)

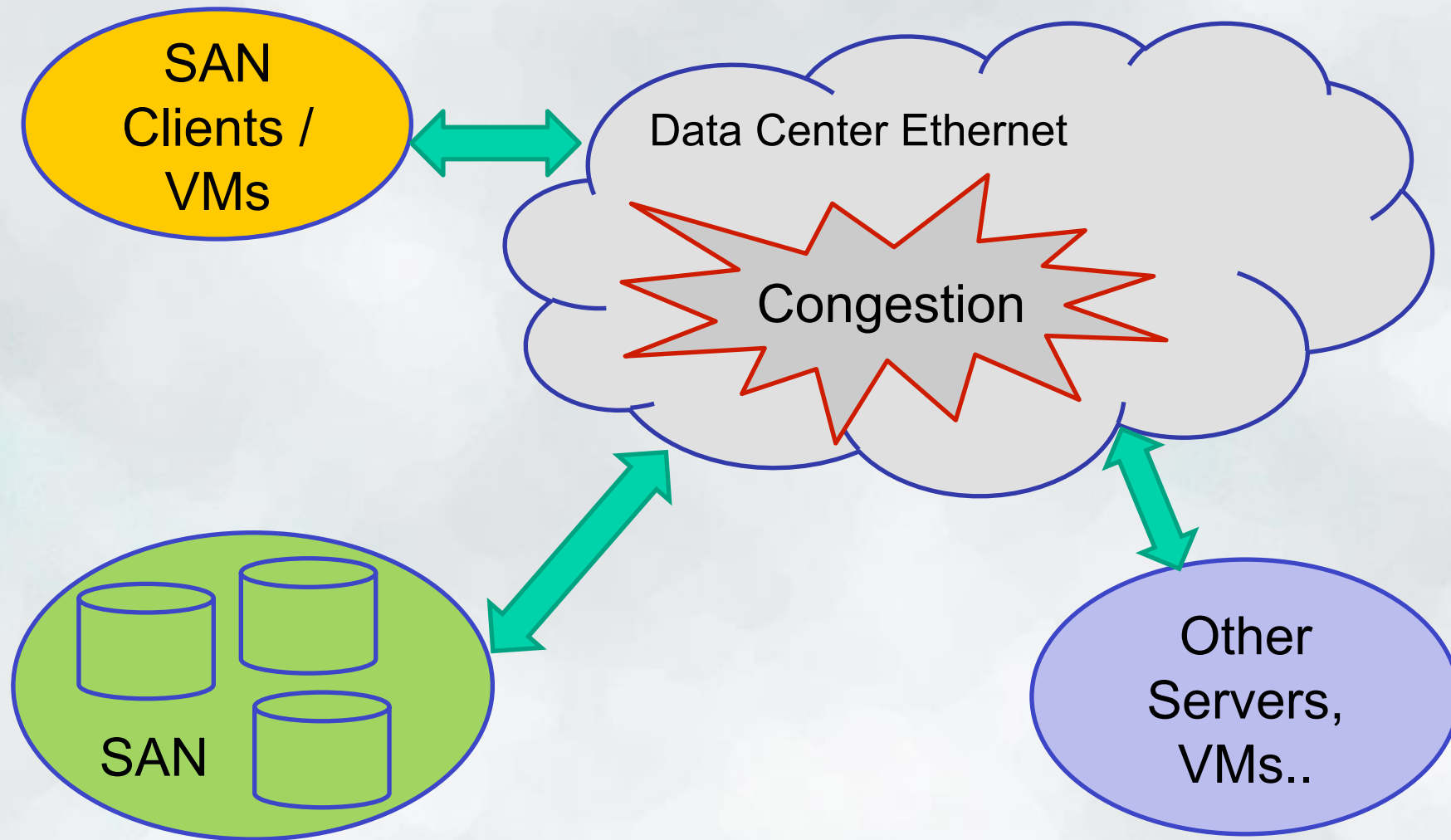
Website: <http://osnet.cs.binghamton.edu/projects/xco.html>

# Outline of the talk

- Motivation & Problem Statement
- Our Contributions – Explicit coordination
- Illustrating the impact of congestion
- XCo framework
  - Time slice scheduling
  - Implementation
- Evaluation
- Future work
- Conclusion

# Motivation

# Congestion in Data Center Ethernet



iSCSI, FCoE,... Streaming Media... Data Mining,... VM Checkpointing, VM Migration,... Voice over IP..., etc...

***Problem: Increasing congestion in Data Center Fabric***

# Our Contribution

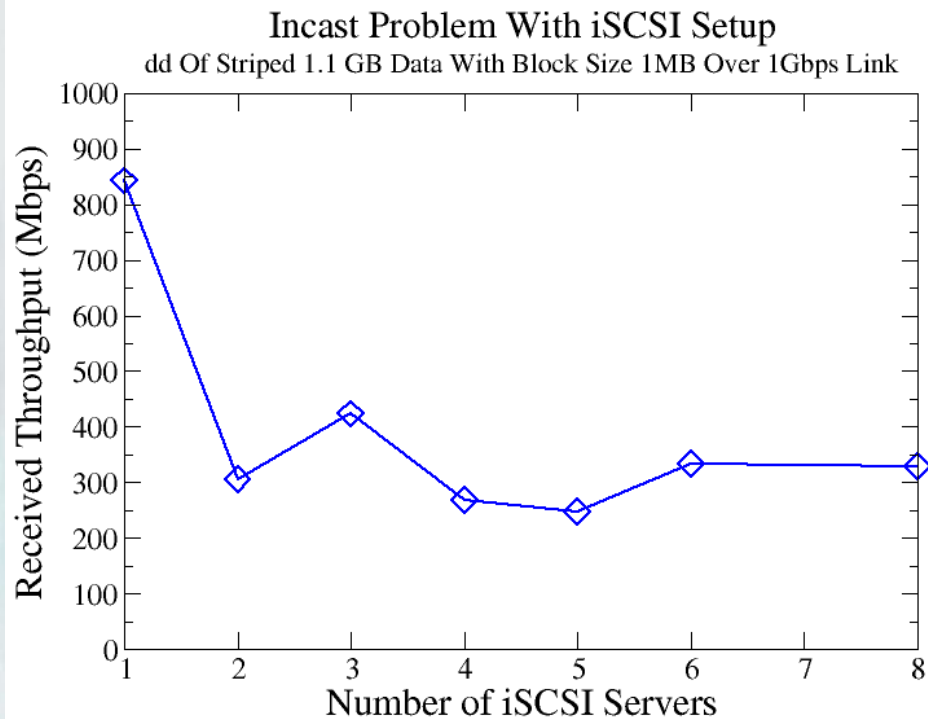
- XCo prevents network fabric congestion
  - By explicitly coordinating the transmissions from end-hosts
- Achieves significantly higher throughput
  - Compared to uncoordinated transmissions
- Completely transparent to the VMs
- Requires no hardware or switch support
- Can be completely implemented on today's commodity Ethernet fabric and switches

# Demonstrating Ethernet Fabric Congestion

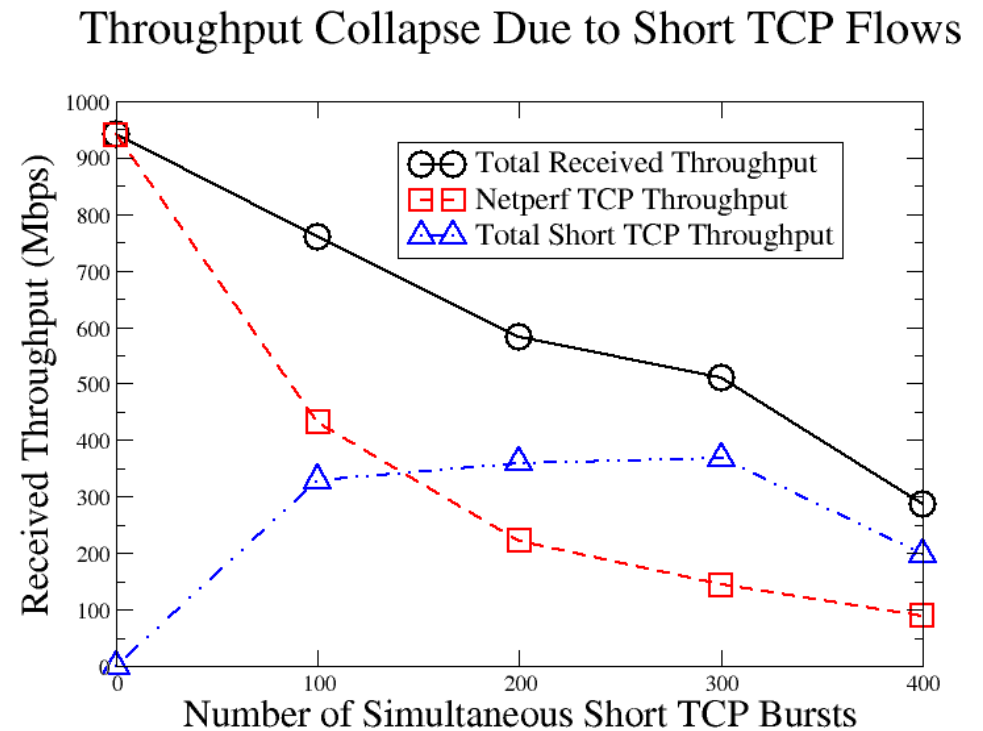
# Experimental Testbed

- Switches: Nortel 4526-GTX
  - 24 1000Base-T ports & 2 XFP 10Gbps optical uplink slots
- Hosts: Xen 3.3.1/Linux 2.6.29.2, Dual CPU Quad Core
- Benchmarks: Netperf, Open-iSCSI, ShortTCP

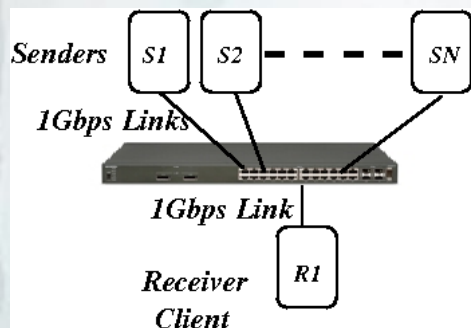
# Incast & Short TCP Flows Problem



(a)

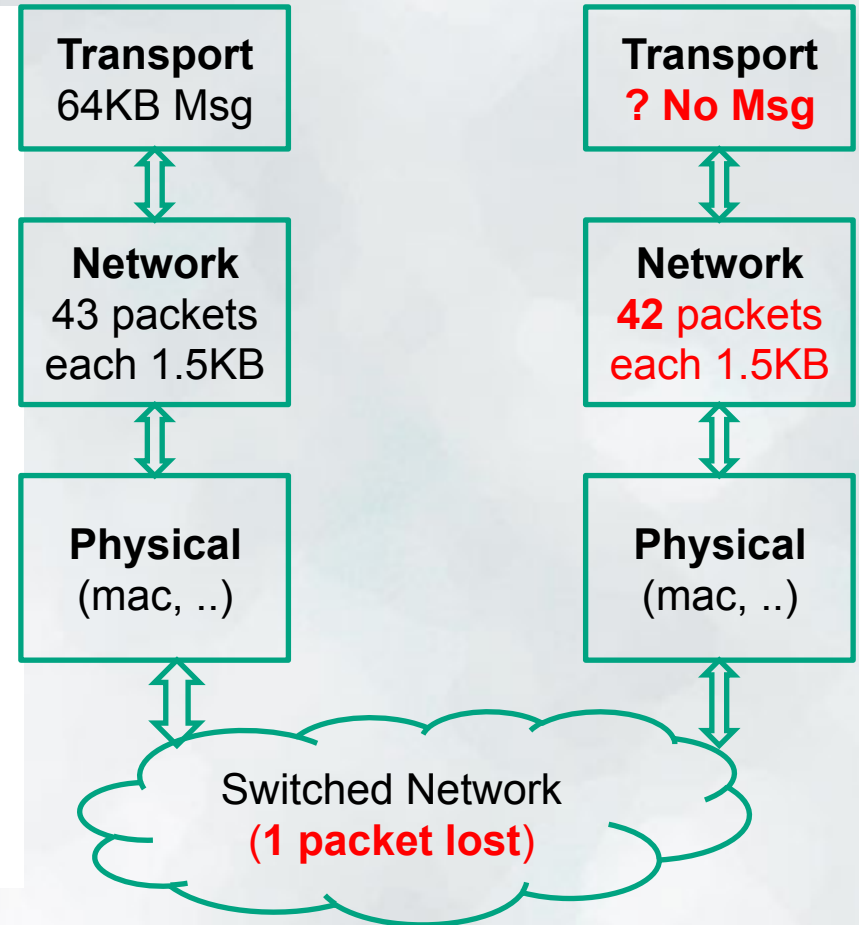
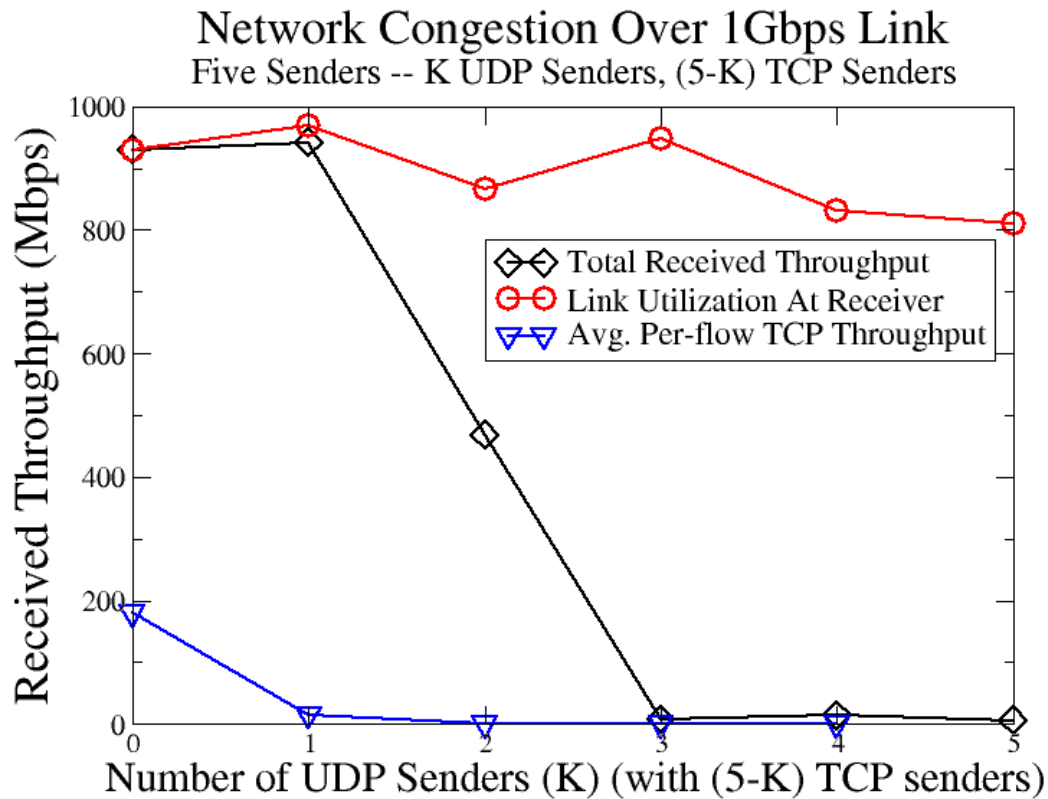


(b)

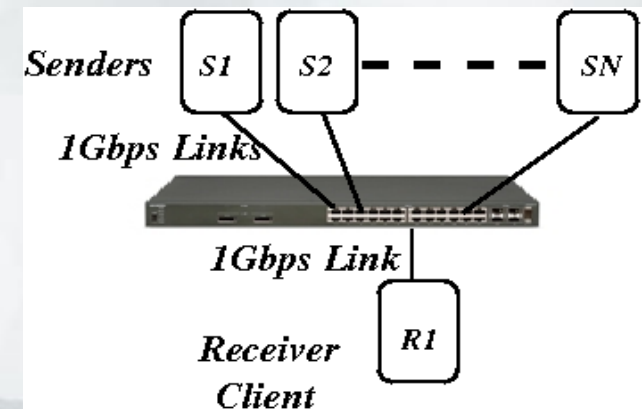


- a) Incast effect: Collapse of TCP throughput due to barrier synchronized traffic
- b) Short TCP flows behave like UDP

# Impact of Non-TCP traffic

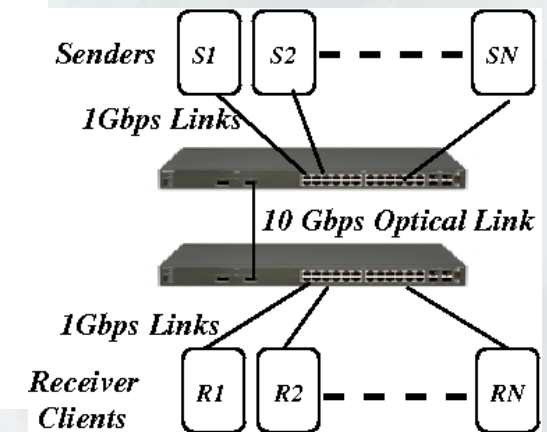
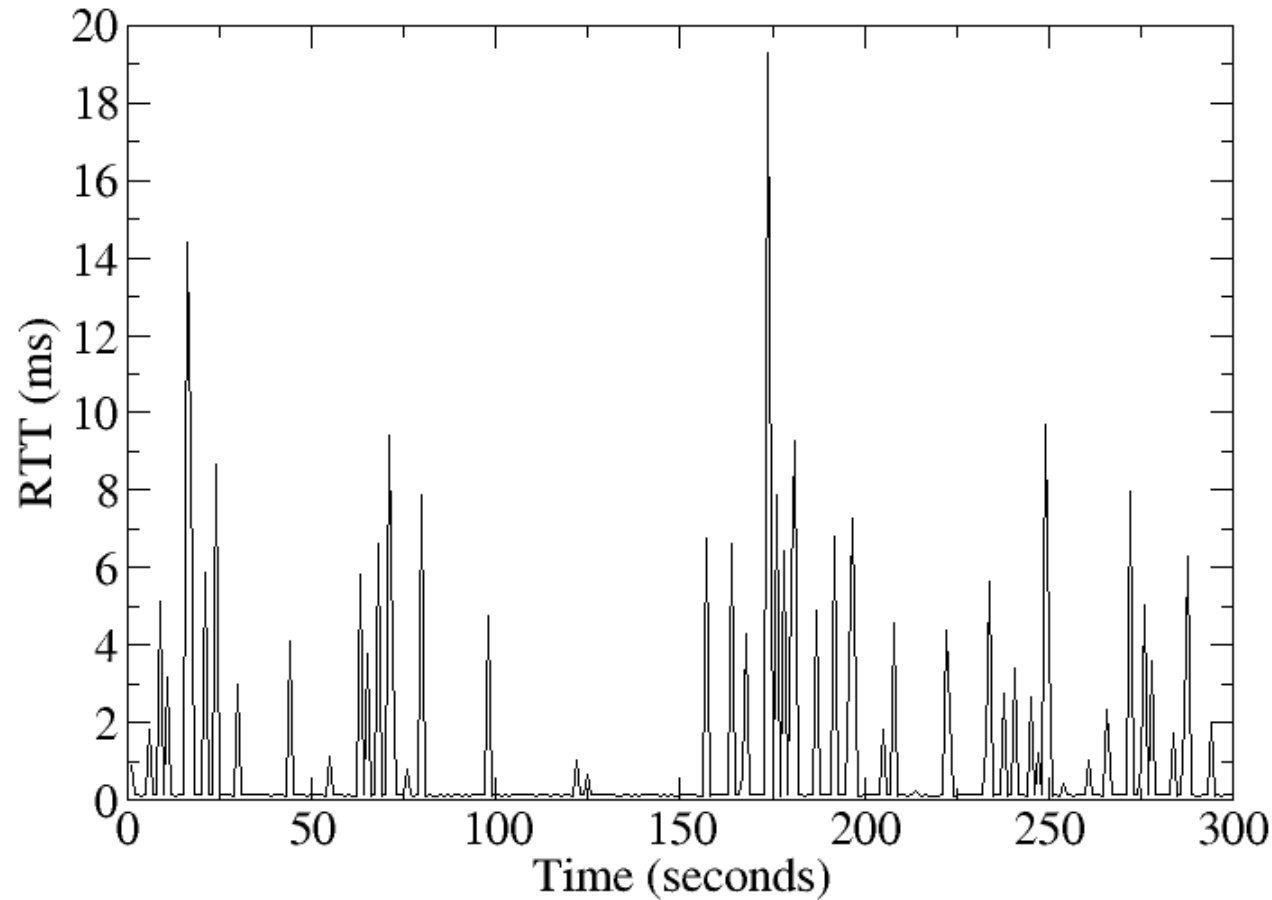


- Bottleneck link is completely utilized
- Total throughput suffers due to large UDP message size
- TCP does not get its fair share due to congestion



# Variation in RTT due to congestion

ping RTT Fluctuation During Congestion



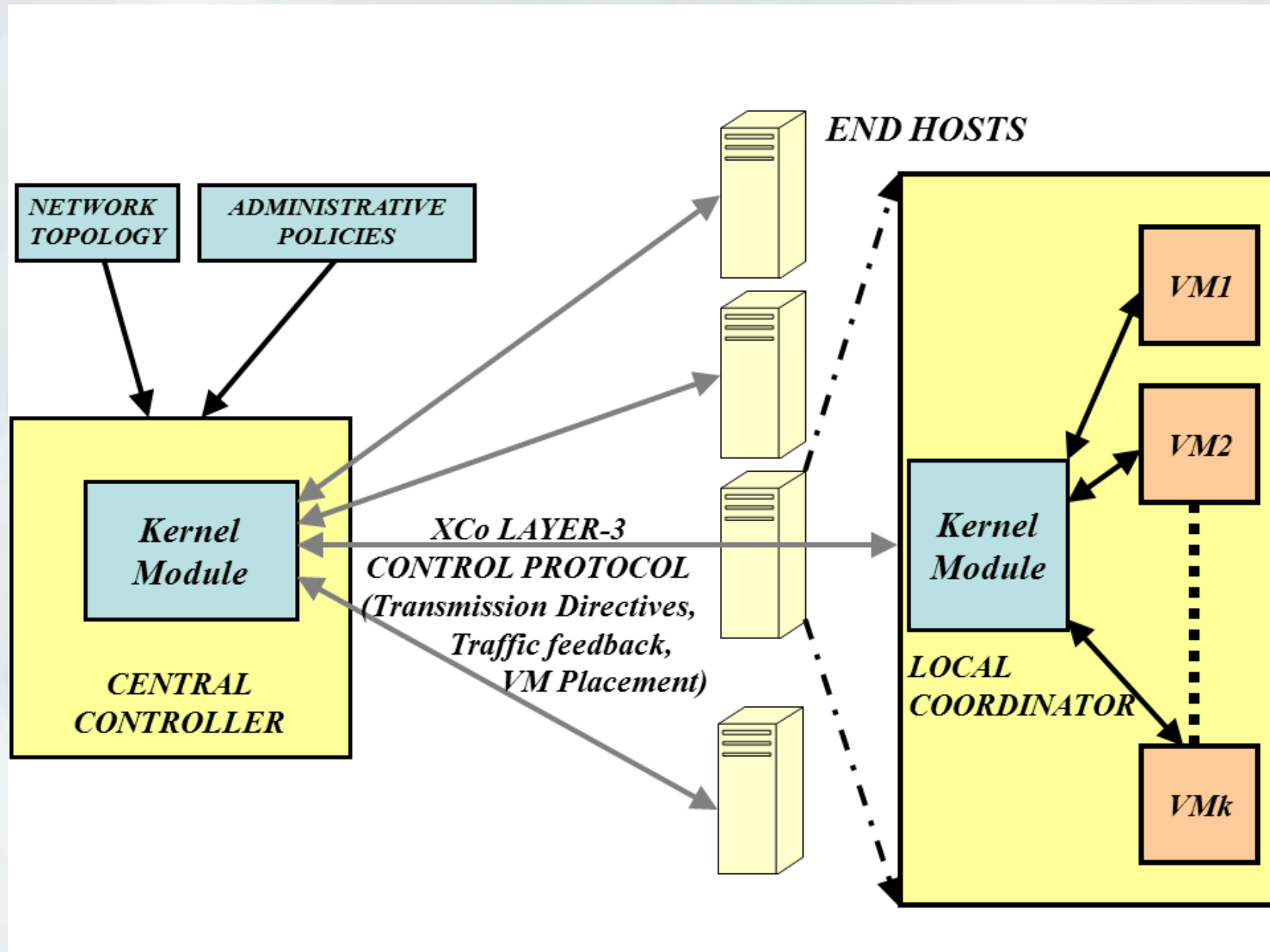
Ping RTT varies significantly due to congestion in the bottleneck link

**Our Solution: XCo**

# Design Goals

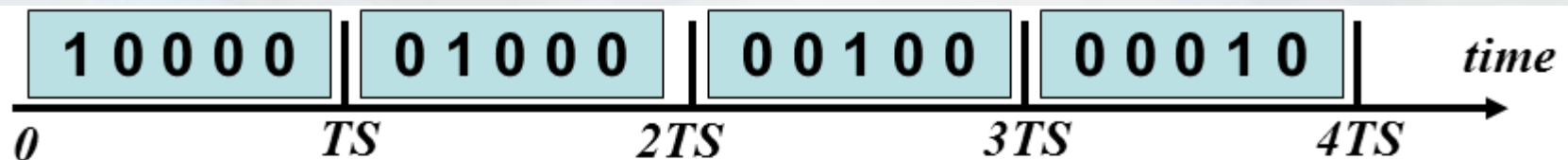
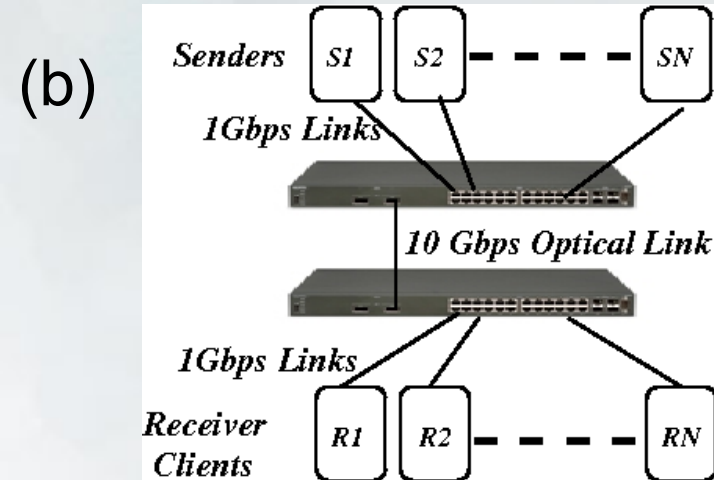
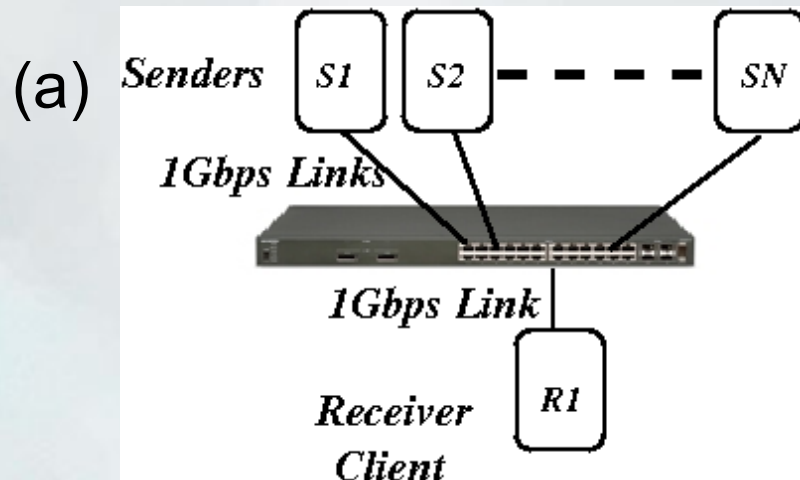
- Prevent excessive concurrent transmissions
  - To avoid potential congestion in bottleneck links of the switched network
- Permit sufficient network activity
  - To achieve high network utilization

# XCo Architecture

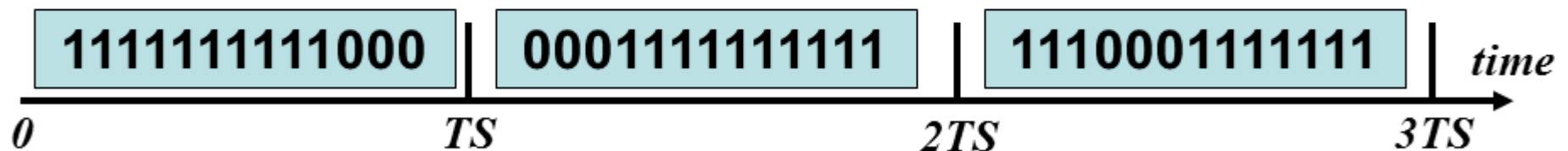


# Transmission Directive

- Permit just enough senders to transmit so as to saturate the bottleneck link

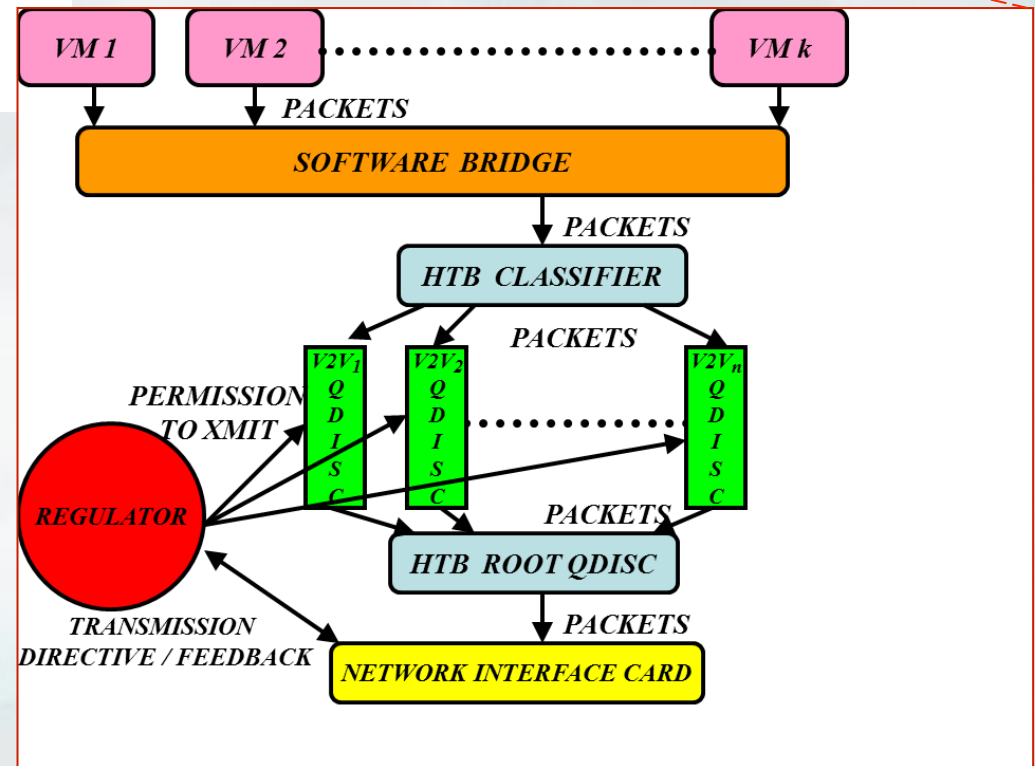
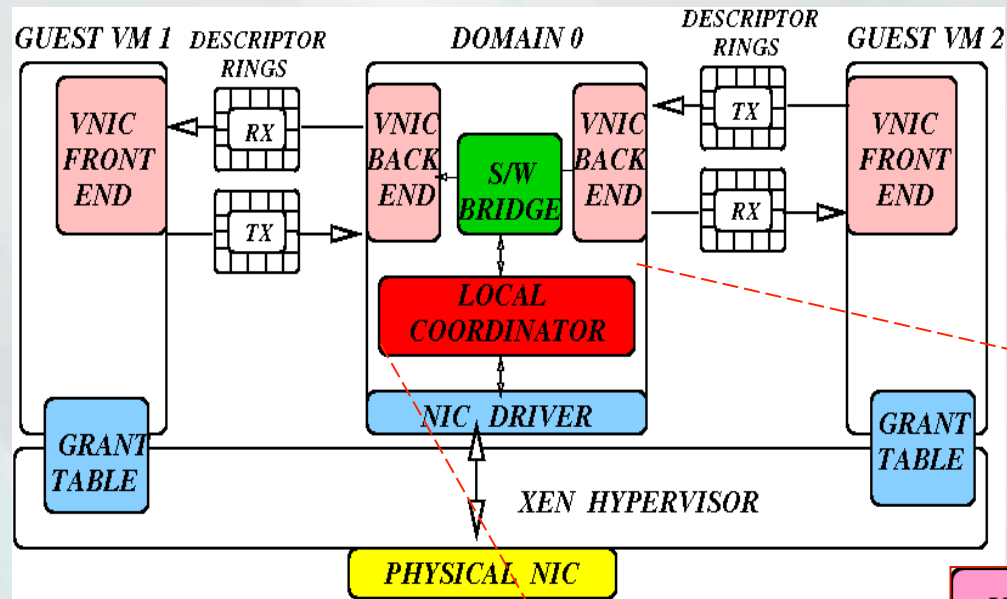


(a) Timeslice scheduling for 1Gig link



(b) Timeslice scheduling for 10Gig link

# XCo Design and Implementation



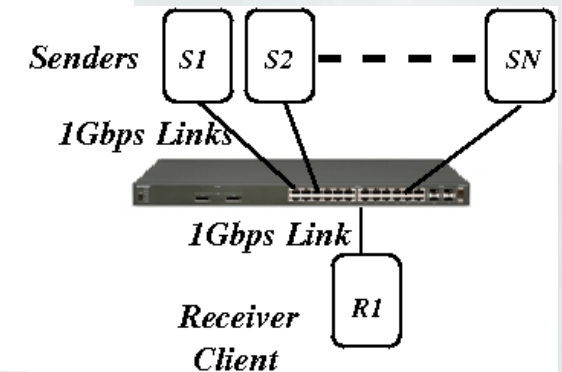
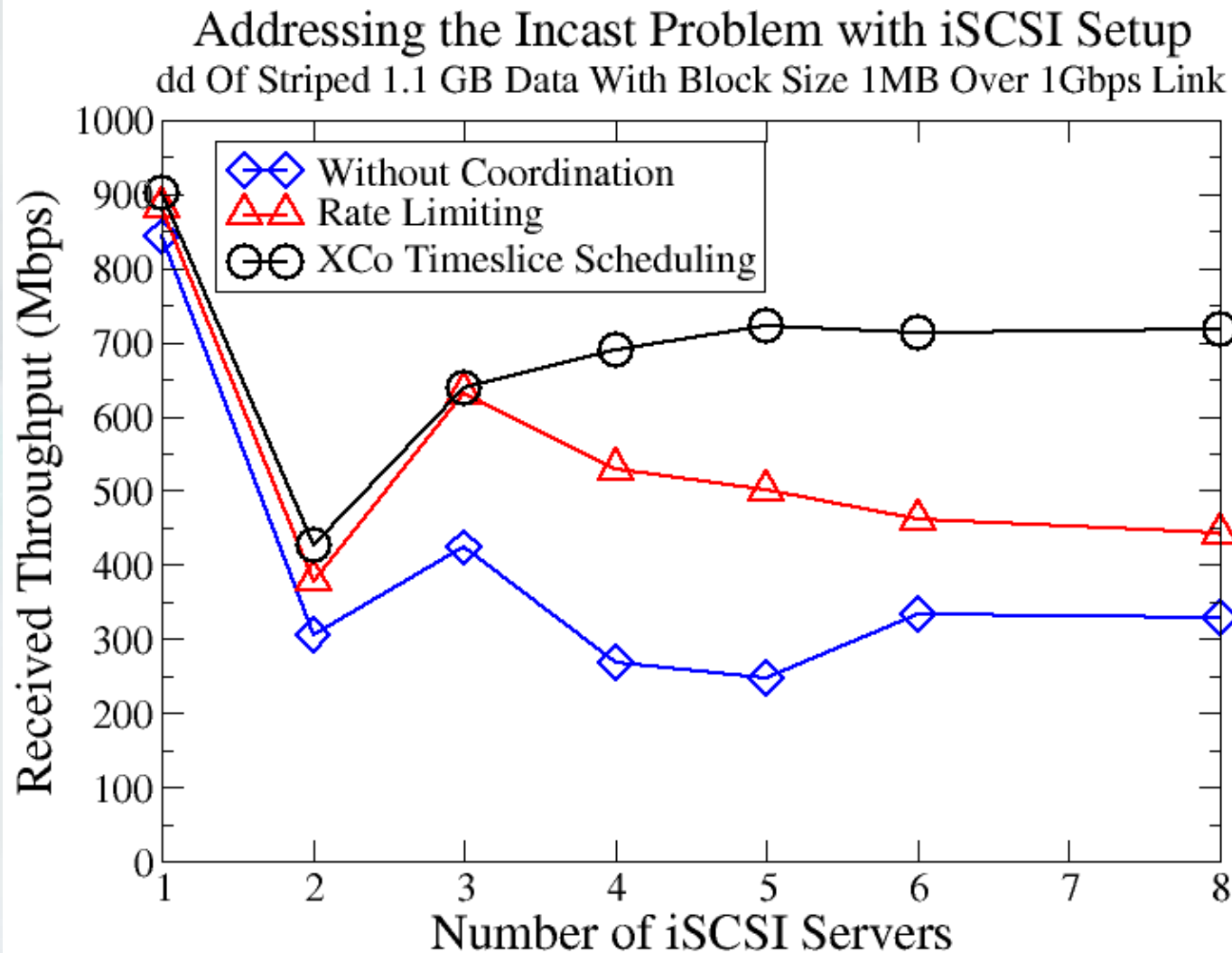
# Timeslice Scheduling For General Network Topology

- The timeslice scheduling algorithm runs in the central controller
- The algorithm considers
  - Multiple bottleneck links
  - Work conservation
  - Some level of fairness among senders
- Feasibility condition for bottleneck link:

$$F(\alpha_{ij}, C_{ij}) : \sum_{\forall x \in \alpha_{ij}} C_x \leq C_{ij}$$

# Performance Evaluation

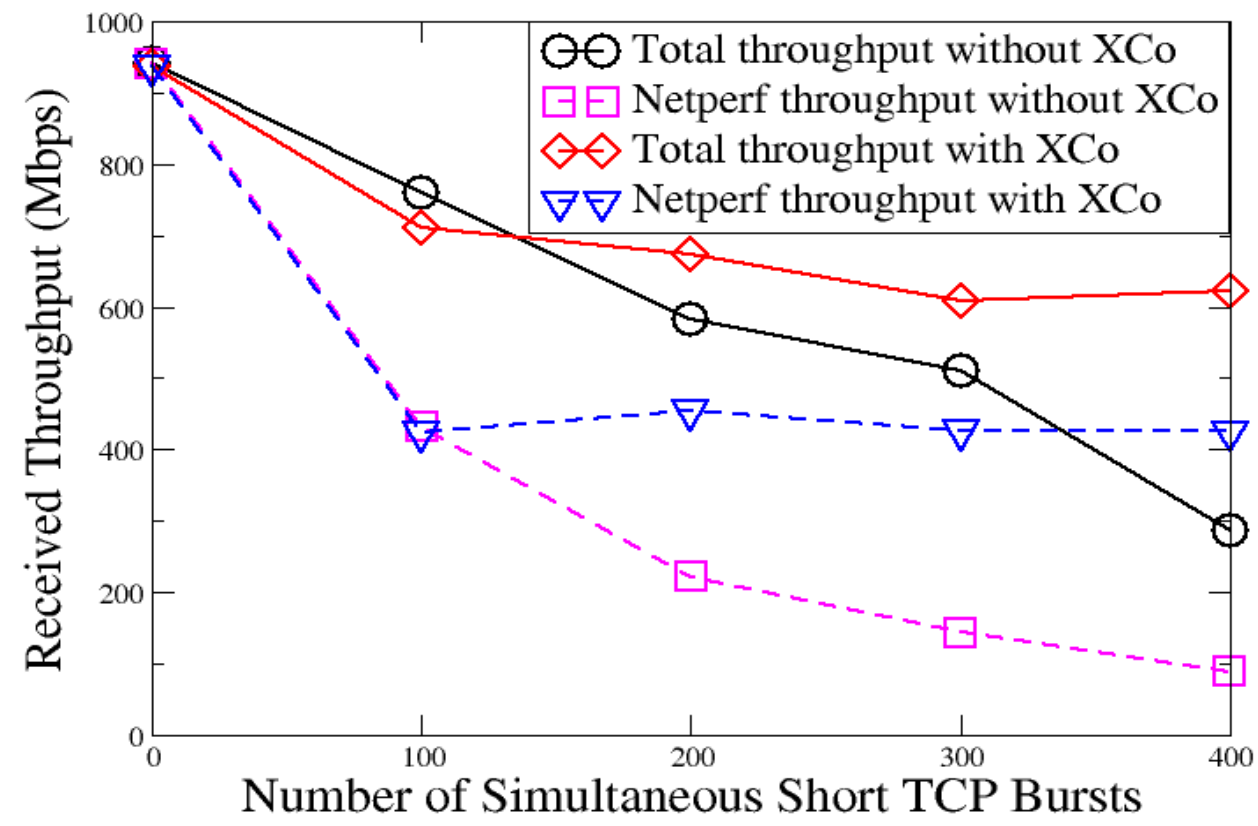
# Solving the Incast Problem using XCo



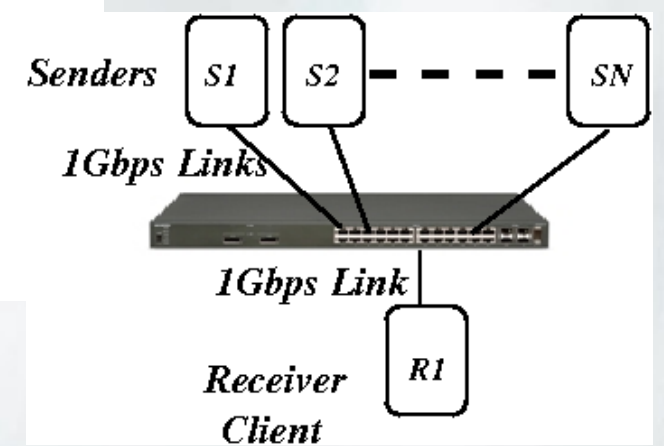
- Incast is alleviated for sufficiently large number of iSCSI servers
- iSCSI throughput improves by 120%

# Solving Throughput Collapse Due To Short TCP Bursts using XCo

Improving Throughput With Short TCP Flows

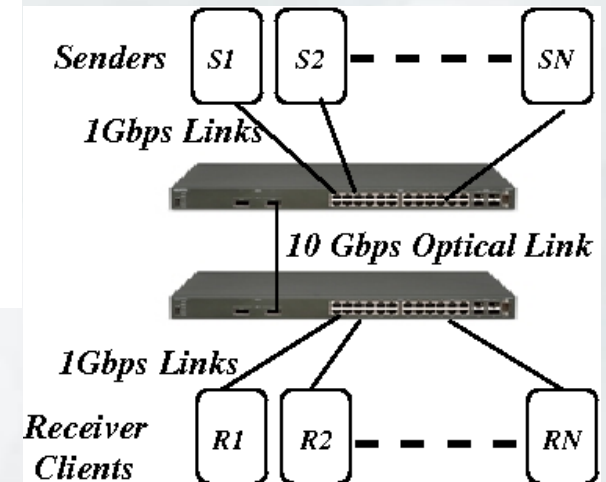
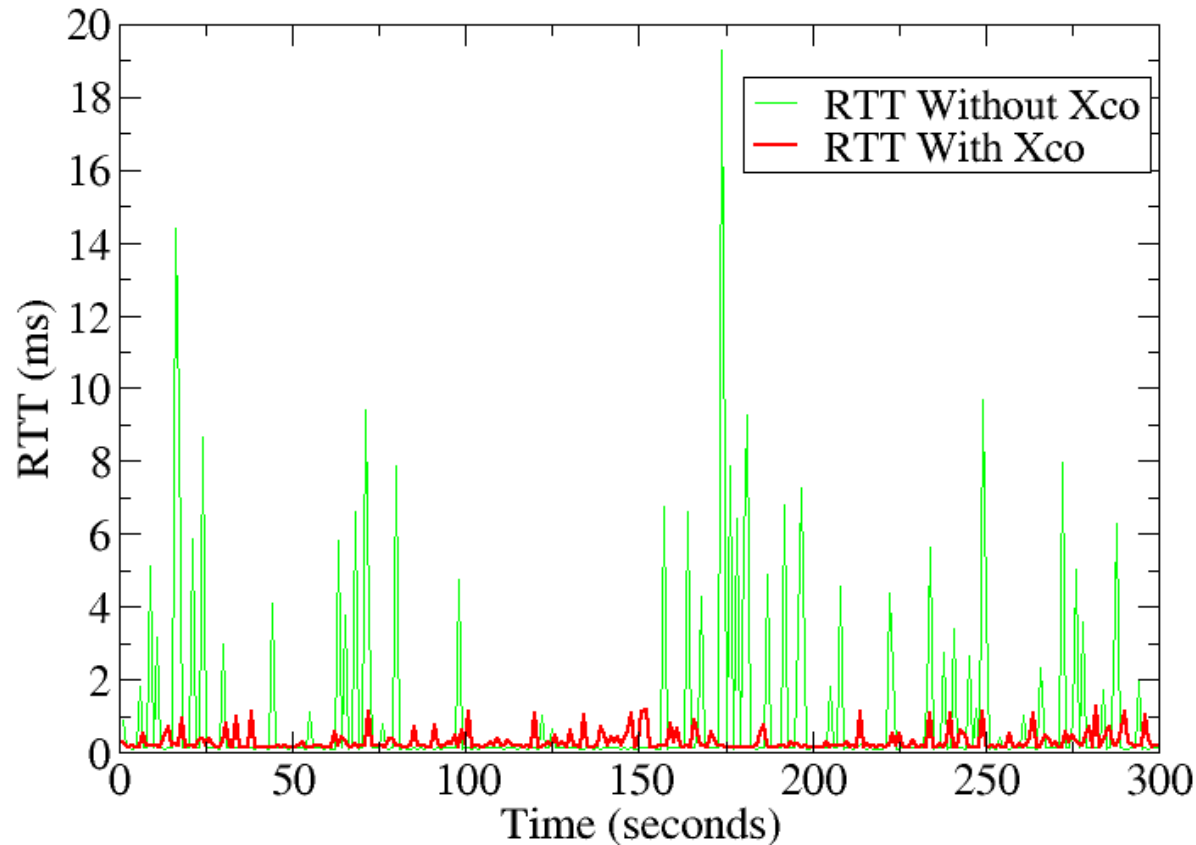


TCP netperf improves up to 320%  
Total throughput improves up to 110%

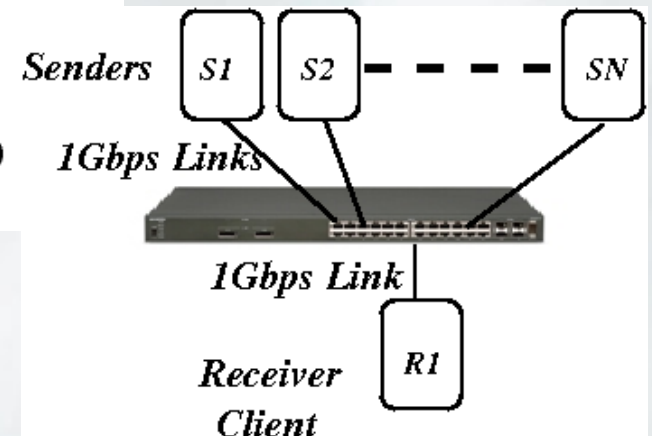
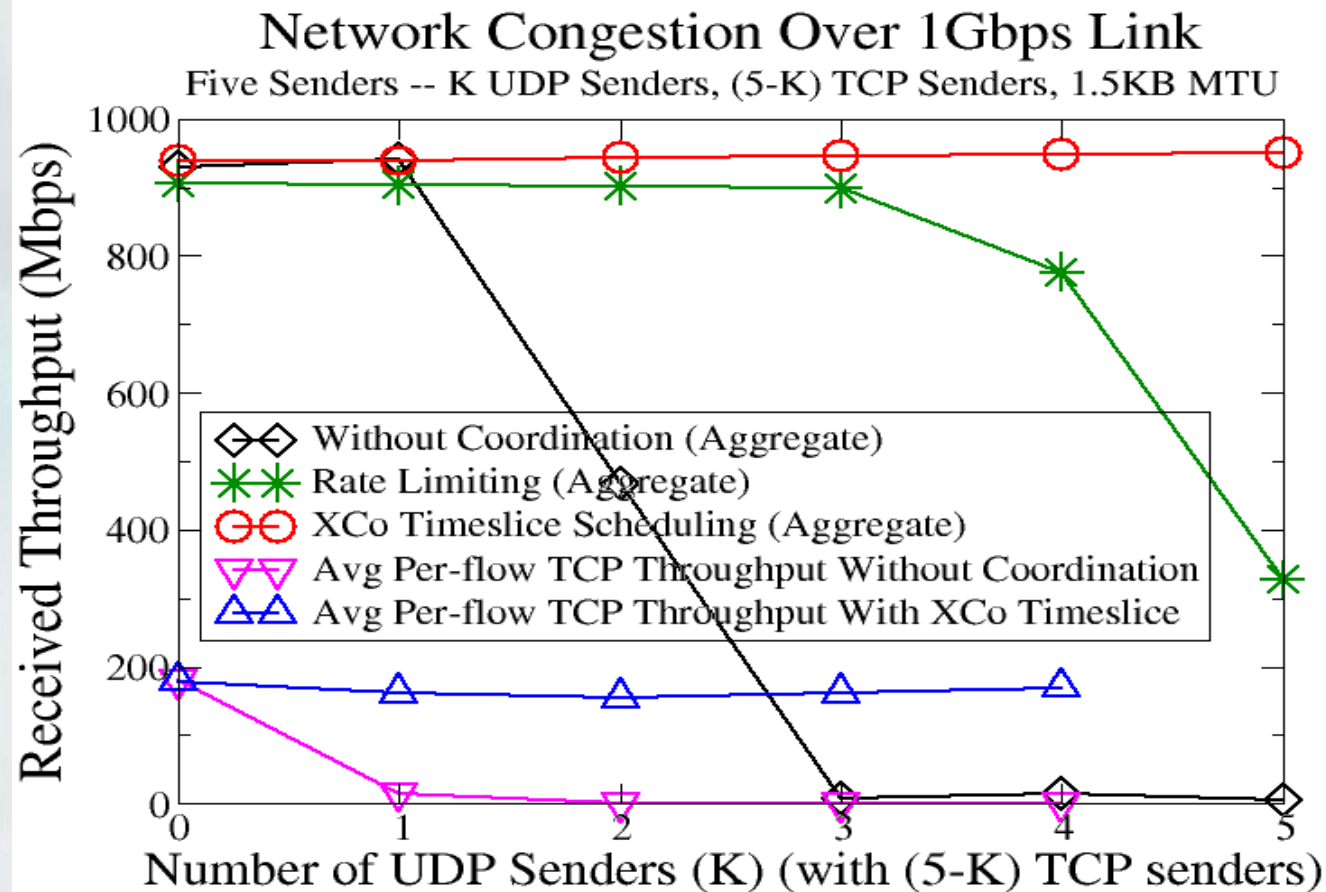


# Decreasing RTT variations with XCo

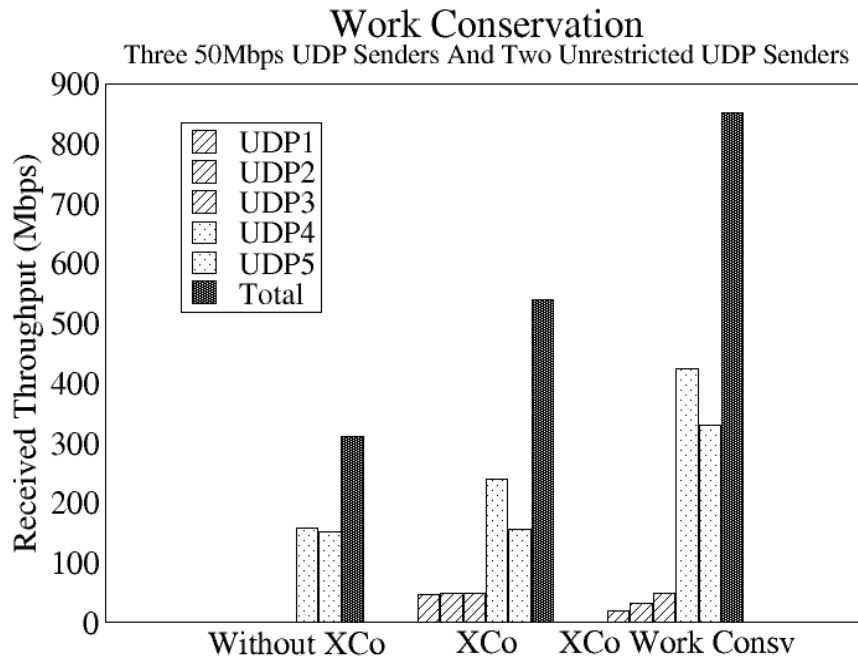
ping RTT Under Congestion



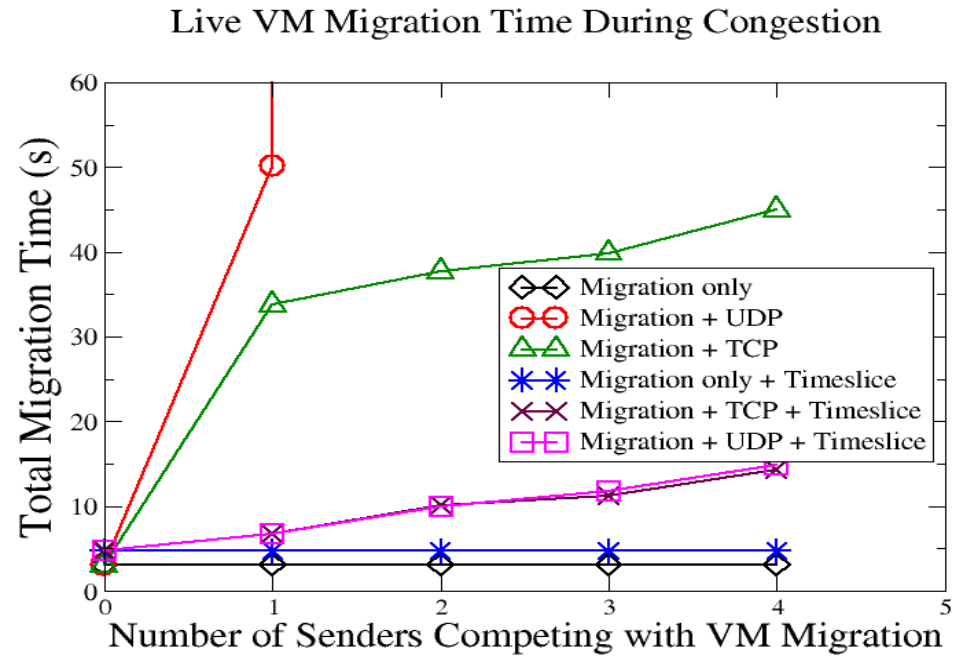
# Preventing Throughput Collapse with Non-TCP Traffic



- No throughput collapse with Xco
- TCP gets its fair share



(a)



(b)

- (a) Total throughput improves with XCo & with Work-conservation
- (b) XCo reduces live VM Migration time in the presence of competing TCP/UDP traffic

# Related Work

- 802.3x Pause Frames
  - Cause head of the line blocking
- VLANs
  - Do not provide congestion control
- Data Center Bridging (DCB) Task Group
  - In progress; future switch-level support for congestion control and QoS
- Hedera, Viking, SPAIN, Ethane, Fat-Tree
  - Depend on modifying switch forwarding tables
  - Reconfiguring spanning trees using VLANs/Multi-pathing
  - For flow scheduling in enterprise networks
    - XCo doesn't need to modify switches or VLANs
- TCP Throughput Collapse (Incast)
  - Reducing concurrent iSCSI servers
  - Reducing advertised TCP recv buffer size
  - Reducing RTOMin

XCo can work with unmodified TCP stack and any number of iSCSI servers

# Future Work

- Empirical Studies
  - More detailed studies on complex topologies with multiple bottlenecks
- Scalability
  - Multiple or hierarchical controllers. NS3 simulations for thousands of nodes under multi-tiered multi-rooted data center topologies
- Active and On-demand Coordination
  - Use feedbacks to trigger central coordination only during times of network congestion
- Fault Tolerance
  - Controller failure, end-host failure
- Alternative Coordination Strategies
  - Hybrid of timeslice scheduling and rate-limiting

# Conclusion

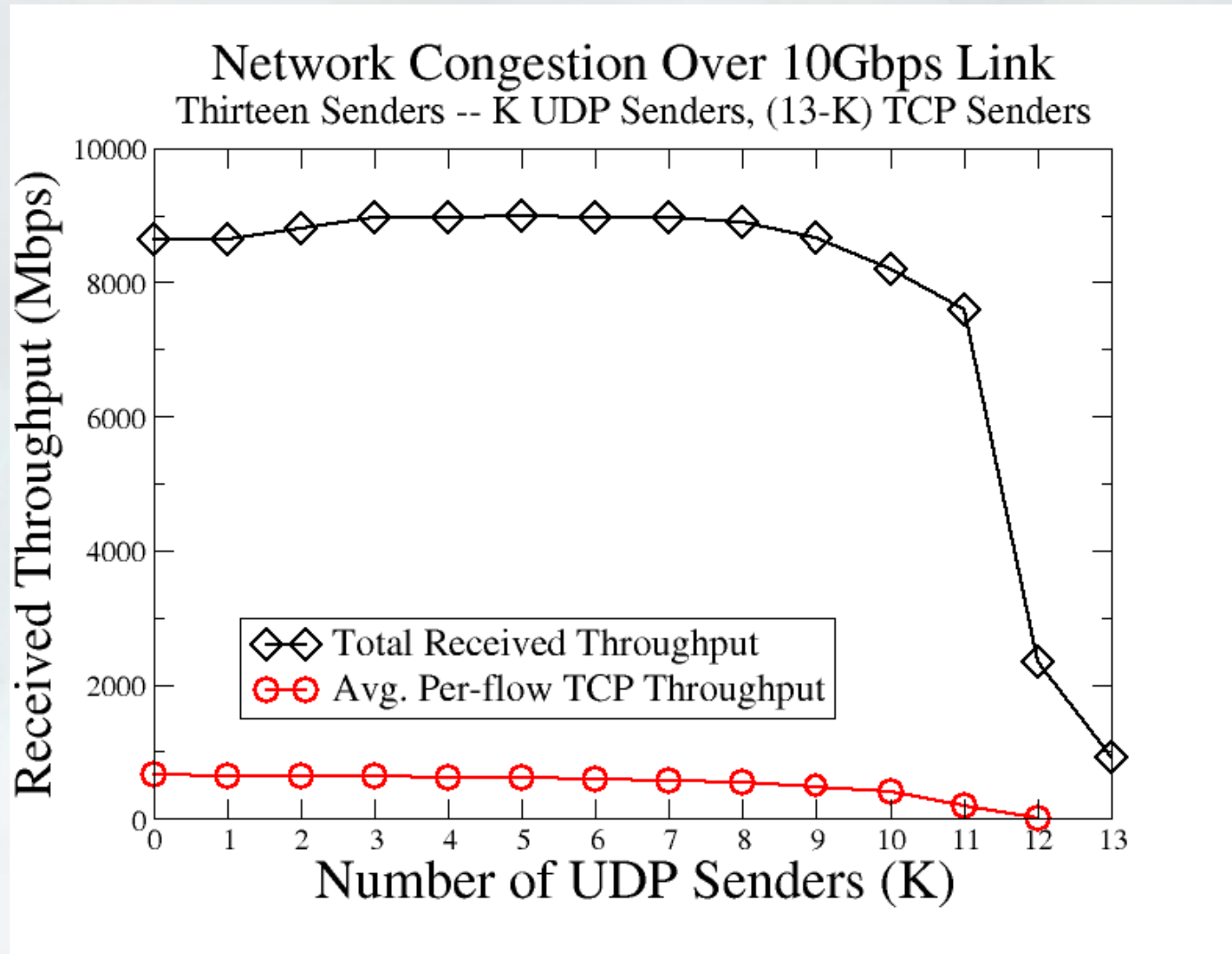
- Virtualization offers new opportunities for mitigating Data Center Ethernet Congestion
- XCo: A system to explicit coordinate the network transmissions from multiple senders at millisecond granularity
- No changes to the VMs, applications, protocols & switches
- A central controller issues transmission directives (or permissions to transmit) to end-hosts
- Temporally separates transmissions competing for bottleneck links
- XCo has the potential to prevent congestion-induced performance problems in today's unmodified Gigabit and 10GigE switched Ethernet
- Future work: more complex topologies, scalability, on-demand coordination, fault-tolerance, and alternative coordination strategies

?

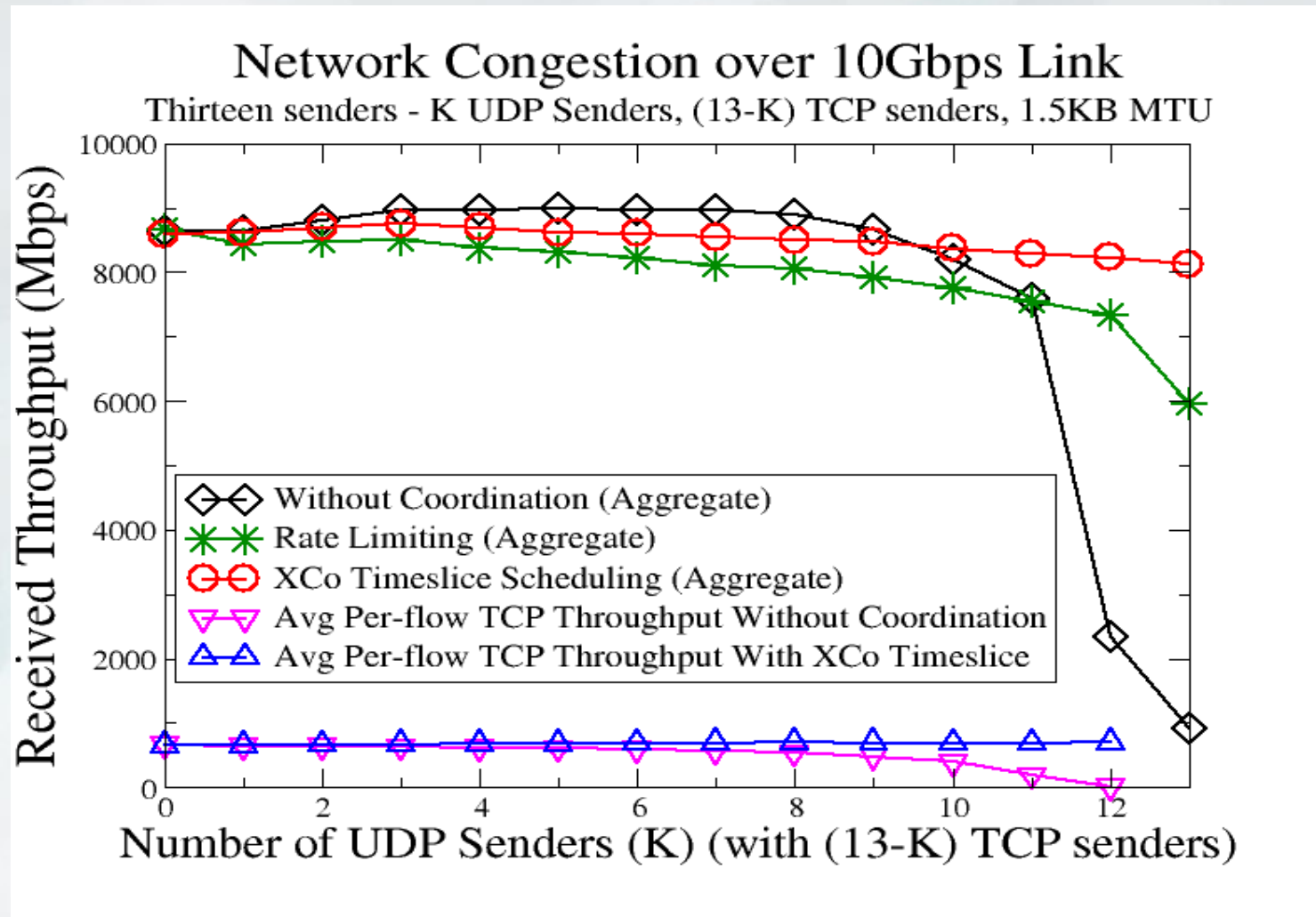
**Thank You!**

Backup

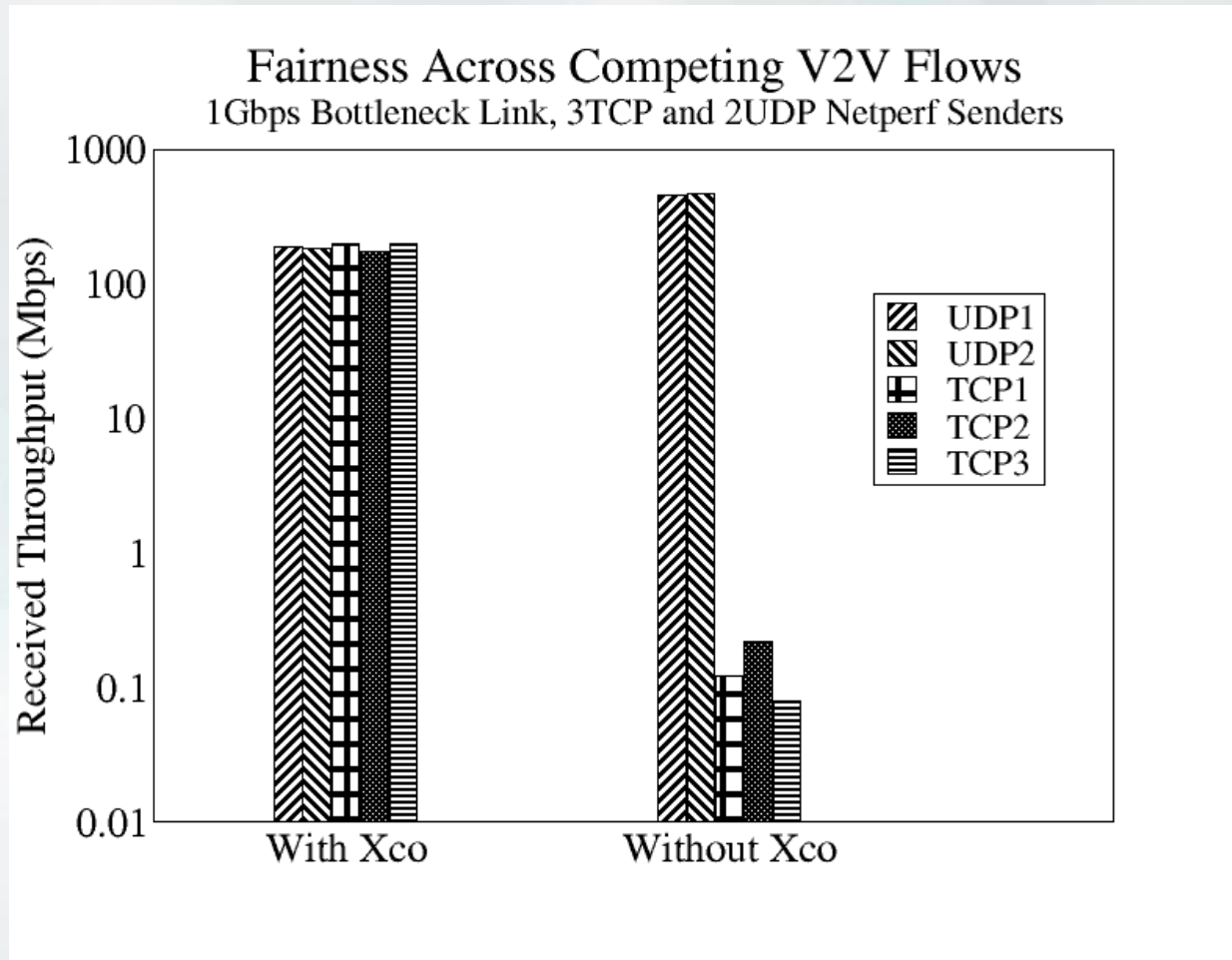
# Throughput Collapse with 10Gig Link



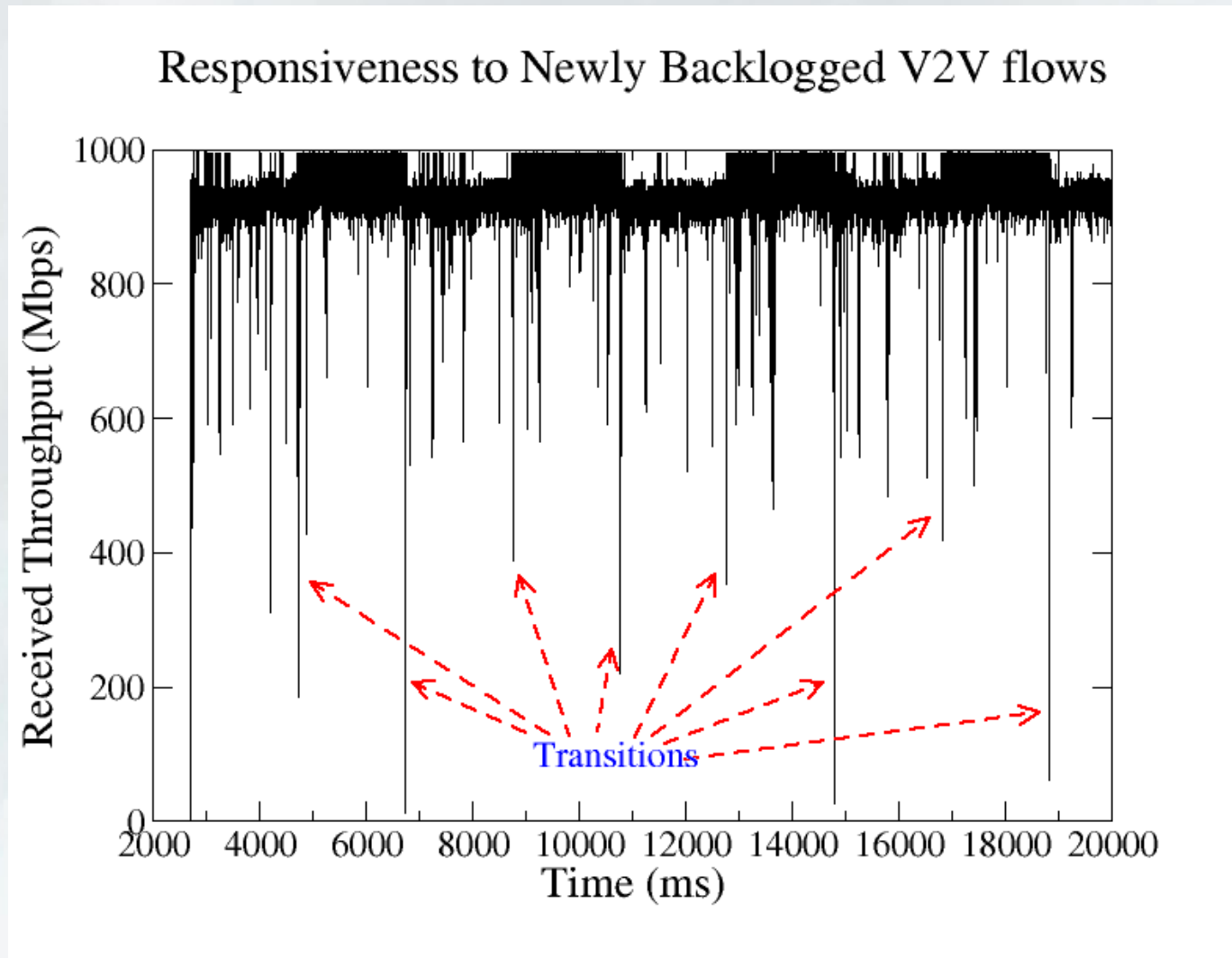
# Solving 10Gig Link Throughput Collapse



# Fairness With Different Flows Achieved By XCo



# XCo Responsiveness To Flow Changes



Responsiveness is about 75ms one time to create new flow classifiers and about 10ms each time when senders change

# XCo Algorithm

$$P_{sd} = \{ (i, j) \mid (i, j) \text{ lies in the path from } s \text{ to } d \text{ in the spanning tree } ST \}$$

$$\beta_{ij} = \{ s \mid \exists d \text{ where } d \in B_s \text{ and } (i, j) \in P_{sd} \}$$

$$\alpha_{ij} = \{ s \mid D_s \neq \text{null and } (i, j) \in P_{sD_s} \}$$

$$F(\alpha_{ij}, C_{ij}) : \sum_{\forall x \in \alpha_{ij}} C_x \leq C_{ij}$$

# XCo Algorithm

```
1: Input: (a) Current spanning tree  $ST$ 
2:   (b) Maximum link capacity  $C_{ij} \forall (i,j) \in ST$ 
3:   (c) Pre-computed paths  $P_{xy} \forall$  nodes  $x, y$ 
4:   (d) Current backlog group  $B_x \forall$  nodes  $x$ 
5:   (e) Current contention group  $\beta_{ij} \forall$  links  $(i, j)$ 
6:   (f) Current active contention group  $\alpha_{ij} \forall$  links  $(i, j)$ 
7:   (g) Last transmission directive  $D_x^{old} \forall$  nodes  $x$ 
8:   (h) Type of scheduling event
9:   (i) Node  $t$  which triggered the scheduling event
10: Output: Next transmission directive  $D_x$  for each node
       $x$  affected by the scheduling event
11:
12:  $A := \emptyset$  /*set of nodes affected by scheduling event*/
13: for each node  $d \in B_t$  do
14:   for each link  $(i, j) \in P_{td}$  do
15:      $A := A \cup \beta_{ij}$ 
16:   end for
17: end for
18: if  $D_t^{old} \neq null$  then
19:   for each link  $(i, j) \in P_{tD_t^{old}}$  do
20:      $\alpha_{ij} := \alpha_{ij} - \{t\}$ 
21:   end for
22:   if (scheduling event = work conservation) then
23:      $B_t := B_t - \{D_t^{old}\}$  /* $t$  has no backlog to  $D_t^{old}$ */
24:   end if
25: end if
```

# XCo Algorithm

```
26:  $N := \emptyset$  /*set of nodes with new schedule*/
27: while  $A \neq \emptyset$  do
28:    $x :=$  some node in  $A$ 
29:    $D_x := null$ 
30:   for each node  $d \in B_x$  do
31:     for each link  $(i, j) \in P_{xd}$  do
32:       /*Check feasibility condition*/
33:       if  $F(\{x\} \cup \alpha_{ij}, C_{ij}) = false$  then
34:         Skip to next  $d$  in line 30
35:       end if
36:     end for
37:
38:      $D_x := d$  /* $d$  satisfies feasibility at each link*/
39:
```

# XCo Algorithm

```
40:   if  $D_x^{old} \neq null$  then
41:     /* $x$  will stop transmitting to  $D_x^{old}$ */
42:     for each link  $(i, j) \in (P_{xD_x^{old}} - P_{xD_x})$  do
43:        $\alpha_{ij} := \alpha_{ij} - \{x\}$ 
44:        $A := A \cup (\beta_{ij} - N)$  /*more nodes affected*/
45:     end for
46:   end if
47:   for each link  $(i, j) \in P_{x_d}$  do
48:      $\alpha_{ij} := \alpha_{ij} \cup \{x\}$ 
49:   end for
50:   break;
51: end for
52:  $A := A - \{x\}$ 
53: if  $D_x \neq null$  then
54:   /*newly scheduled; don't reschedule again*/
55:    $N := N \cup x$ 
56: end if
57: end while
58: for each  $x \in N$  do
59:   Send  $D_x$  to  $x$ 
60:    $D_x^{old} := D_x$ 
61: end for
```