

# **97-1085: A Switch Algorithm for ABR Multipoint-to-Point Connections**

**Sonia Fahmy, Raj Jain, Rohit Goyal,  
and Bobby Vandalore**

Department of CIS, The Ohio State University

**Contact:** [jain@cis.ohio-state.edu](mailto:jain@cis.ohio-state.edu)

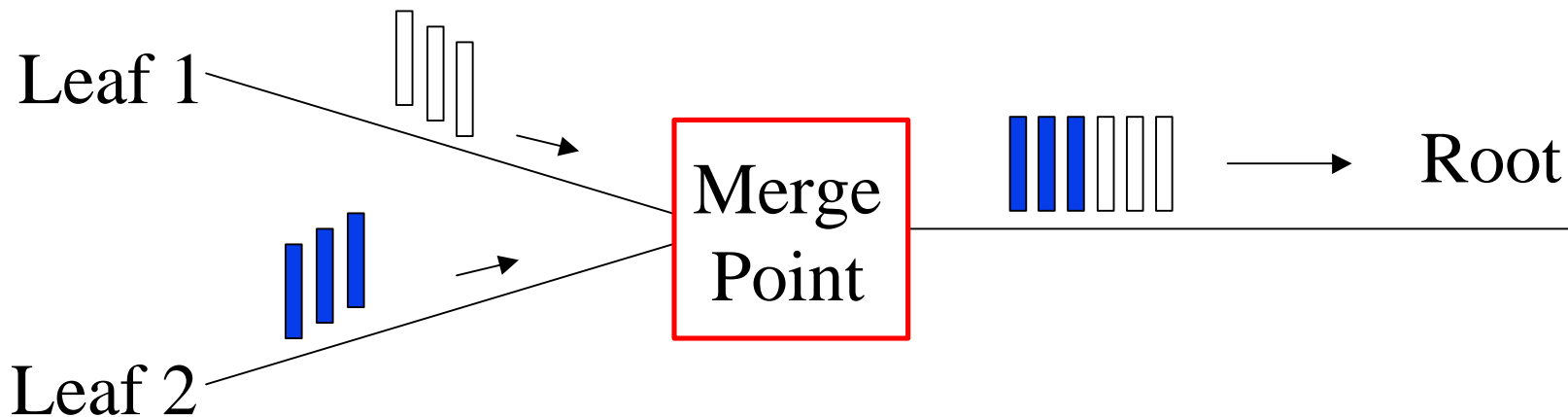
<http://www.cis.ohio-state.edu/~jain/>



- ❑ Multipoint-to-point VCs
- ❑ VC Merging
- ❑ Rate Allocation Algorithm
- ❑ Merging Point Algorithm
- ❑ Design Issues
- ❑ Simulation Results

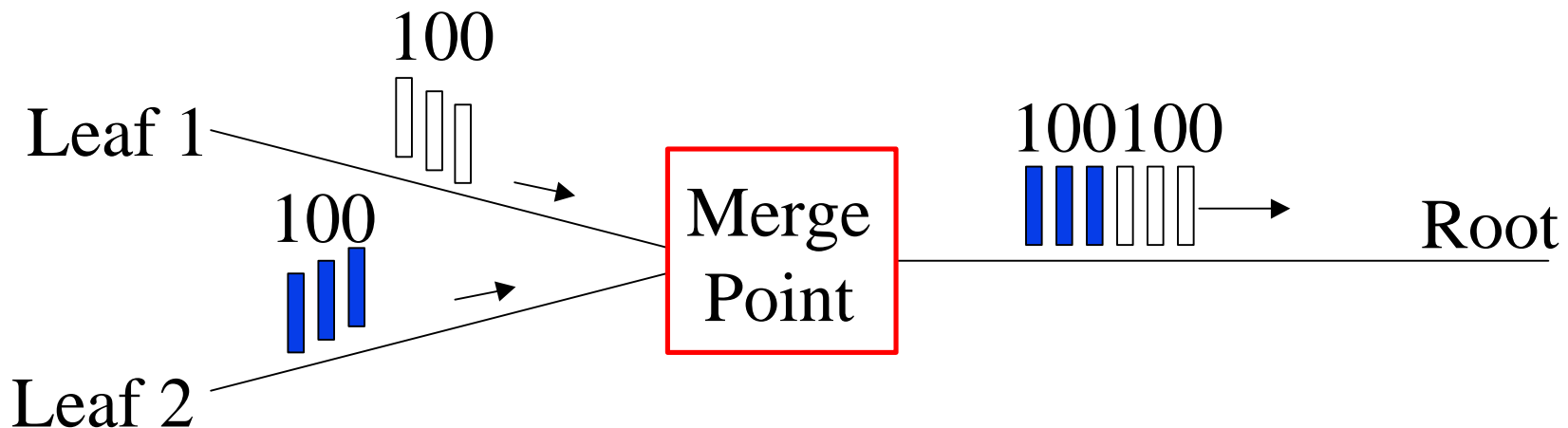
# Multipoint-to-Point VCs

- ❑ More than one concurrent sender
- ❑ Traffic at root  
=  $\Sigma$  traffic originating from leaves
- ❑ Source-based fairness:  
N-to-one connection = N one-to-one connections  
 $\Rightarrow$  max-min fairness among sources



# VC Merging

- ❑ Buffer cells at merging point till EOM bit = 1
- ❑ Cells of senders in the same multipoint-to-point VC cannot be distinguished
- ❑ Question: Can we achieve source-based fairness?



# ERICA+

- ❑ Time is slotted into averaging intervals
- ❑ ABR capacity = [link capacity  
– (VBR + CBR load)] × f(queue length)
- ❑ Estimate input rate =  $\sum CCR_j$
- ❑ overload = input rate/ABR capacity
- ❑ ER<sub>j</sub>\_efficiency = CCR<sub>j</sub>/overload
- ❑ ER\_fairshare = ABR capacity/# of active sources
- ❑ IF overload ≤ 1 + δ THEN ER<sub>j</sub> =  
    max (ER<sub>j</sub>\_efficiency, ER\_fairshare, maxER<sub>previous</sub>)  
    ELSE ER<sub>j</sub> = max(ER<sub>j</sub>\_efficiency, ER\_fairshare)
- ❑ maxER<sub>current</sub> = max(maxER<sub>current</sub>, ER<sub>j</sub>)
- ❑ ER in BRM<sub>j</sub> = min(ER in BRM<sub>j</sub>, ER<sub>j</sub>)

# Changes to ERICA+

- ❑ Remove fair share term (# active sources)
- ❑ Use  $CCR_{jmax}$  instead of  $CCR_j$   
Maximum is calculated in successive intervals
- ❑ To minimize oscillations, use exponential averaging options for:
  - Input rate
  - ABR capacity
  - $maxER_{previous}$

# Rate Allocation Algorithm

**1. FRM cell is received for VC j:**

$CCR_j = CCR \text{ from FRM}$

OR:

IF FirstFRM<sub>j</sub> THEN

$CCR_j = CCR \text{ from FRM}$

FirstFRM<sub>j</sub> = FALSE

ELSE

$CCR_j = \max (CCR \text{ from FRM}, CCR_j)$

## Algorithm (Cont.)

### 2. BRM cell to be sent out for VC j:

IF overload  $> 1 + \delta$  THEN

$$ER = CCR_j / \text{overload}$$

ELSE

$$ER = \max (CCR_j / \text{overload}, \max ER_{\text{previous}})$$

$$ER = \min (\text{capacity}, ER)$$

$$\max ER_{\text{current}} = \max (ER, \max ER_{\text{current}})$$

$$ER \text{ in BRM} = \min (ER, ER \text{ in BRM})$$

[note:  $\delta = 0.1$  in our simulations]

# Algorithm (Cont.)

## 3. End of averaging interval:

input rate = exponential average

capacity = exponential average scaled by queue function

overload = input rate/capacity

$\forall j$ : FirstFRM<sub>j</sub> = TRUE

maxER<sub>previous</sub> = maxER<sub>current</sub>

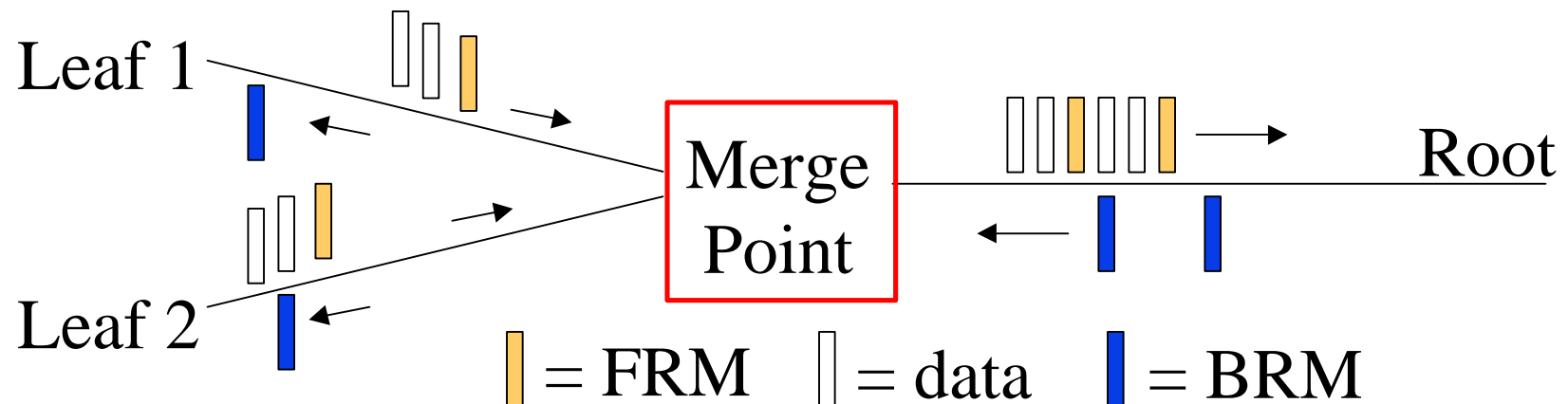
OR: maxER<sub>previous</sub> =  $(1-\alpha) \times \text{maxER}_{\text{current}} + \alpha \times \text{maxER}_{\text{previous}}$

maxER<sub>current</sub> = 0

[note:  $\alpha = 0.1$  in our simulations]

# Merging Point Algorithm

- Maintain a bit at the merging point for each flow being merged  
Bit = 1  
⇒ FRM received from this flow after BRM sent to it
- BRMs are duplicated and sent to flows whose bits are set, then bits are reset



# Design Issues

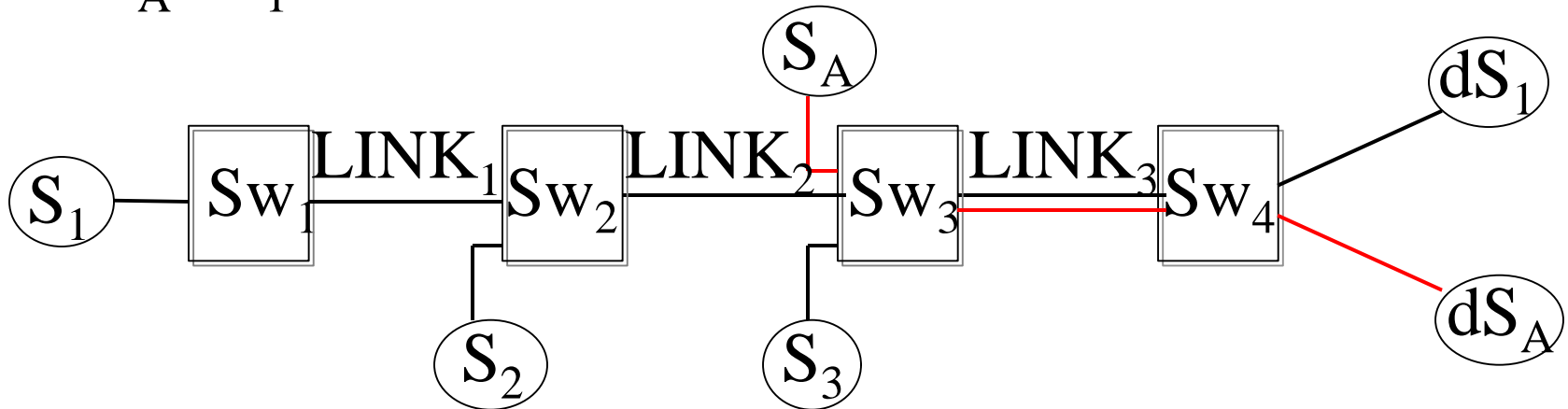
- ❑ Per-source accounting is avoided
- ❑ Using CCR from BRM cells can cause unfairness because CCR in BRM may belong to a source with a higher bottleneck rate than all upstream sources
- ❑ CCR from FRM cells is adequate because of  $\max ER_{\text{previous}}$  term and properties of merging point
- ❑ BRM to FRM ratio at sender and inside the network is close to 1
- ❑ Destinations (not merging points) turn around BRMs for scalability, insensitivity to levels of tree and to avoid noise

# Simulation Parameters

- ❑ Unidirectional traffic
- ❑ RIF = 1/32, 1
- ❑ Rule 6 disabled
- ❑ Queue control:  $a = 1.15$ ,  $b = 1$ , drain limit = 50%, target queuing delay = 1.5 s
- ❑ Measurement interval = 5 ms, 200  $\mu$ s
- ❑ One cell long packets (Avoids VC merging issues)
- ❑ Max CCR and averaging maxERprevious used
- ❑ Link lengths in kms: {LINK1, LINK2, LINK3} = {50, 500, 5000}, {5000, 500, 50}

# Downstream Bottleneck

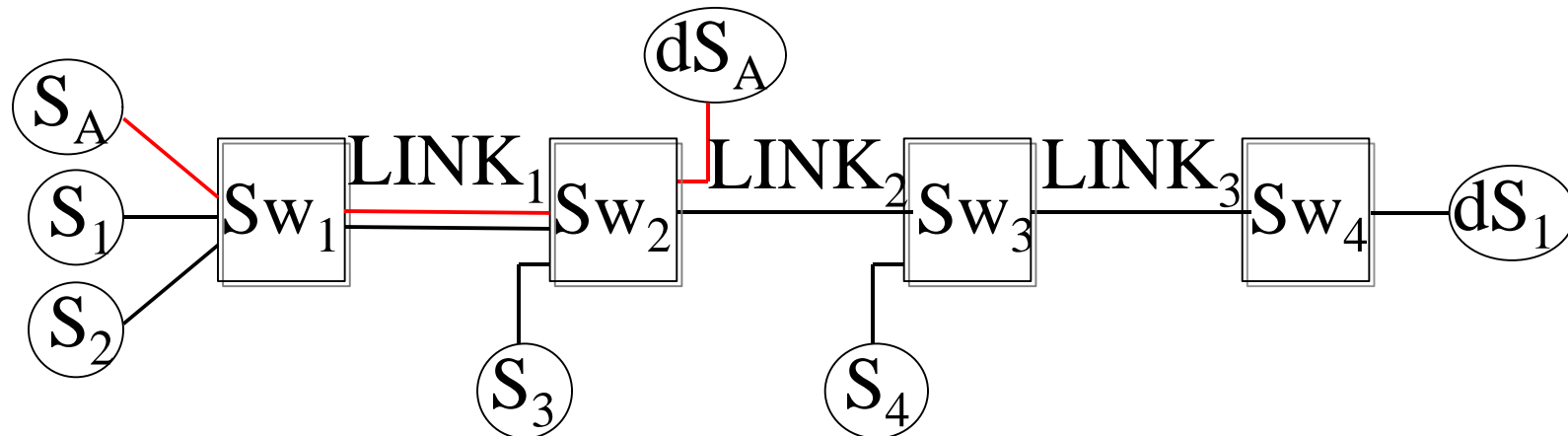
- Goal:  $\{S_1, S_2, S_3, S_A\} \leftarrow \{37.5, 37.5, 37.5, 37.5\}$
- ICRs:  $\{S_1, S_2, S_3, S_A\} \leftarrow \{25, 25, 25, 25\}$ ,  
 $\{100, 100, 100, 100\}, \{65, 10, 65, 10\}$
- **Result:** All sources are allocated 37.5 Mbps, small queues, LINK3 100% utilized, cells received at  $dS_A : dS_1 \approx 175k : 520k \approx 1 : 3$



All links are 150 Mbps

# Upstream Bottleneck

- Goal:  $\{S_1, S_2, S_3, S_4, S_A\}$   
←  $\{16.7, 16.7, 58.3, 58.3, 16.7\}$
- ICRs:  $\{S_1, S_2, S_3, S_4, S_A\}$  ←  $\{20, 20, 30, 80, 10\}$
- Results are similar with different link lengths,  
RIF = 1/32, 1, interval length = 5 ms, 200  $\mu$ s (no RMs  
for  $S_1, S_2, S_A$  for 4 intervals; for  $S_3, S_4$  for 1 interval)

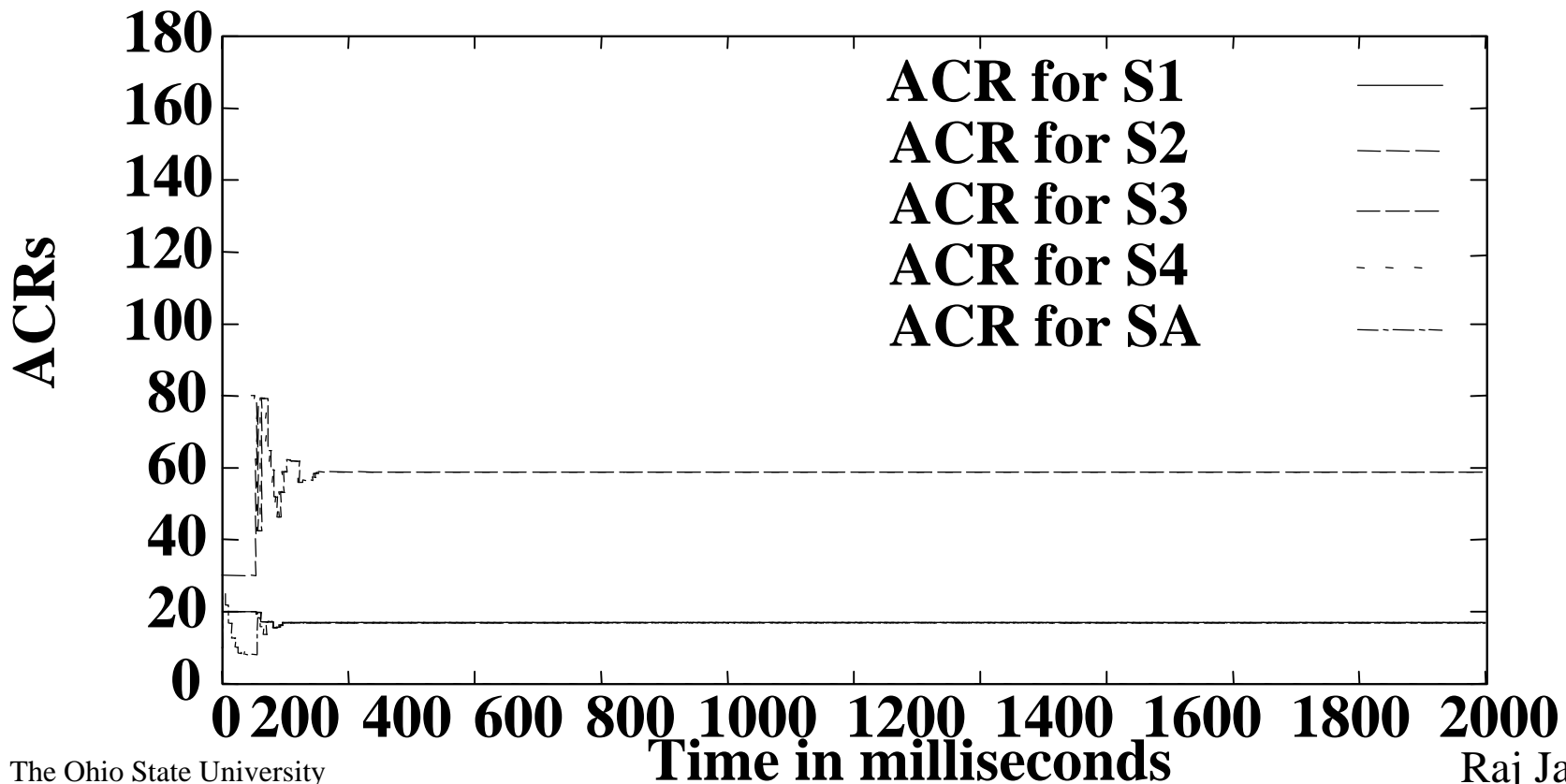


All links are 150 Mbps, except LINK<sub>1</sub> which is 50 Mbps

# Simulation Results

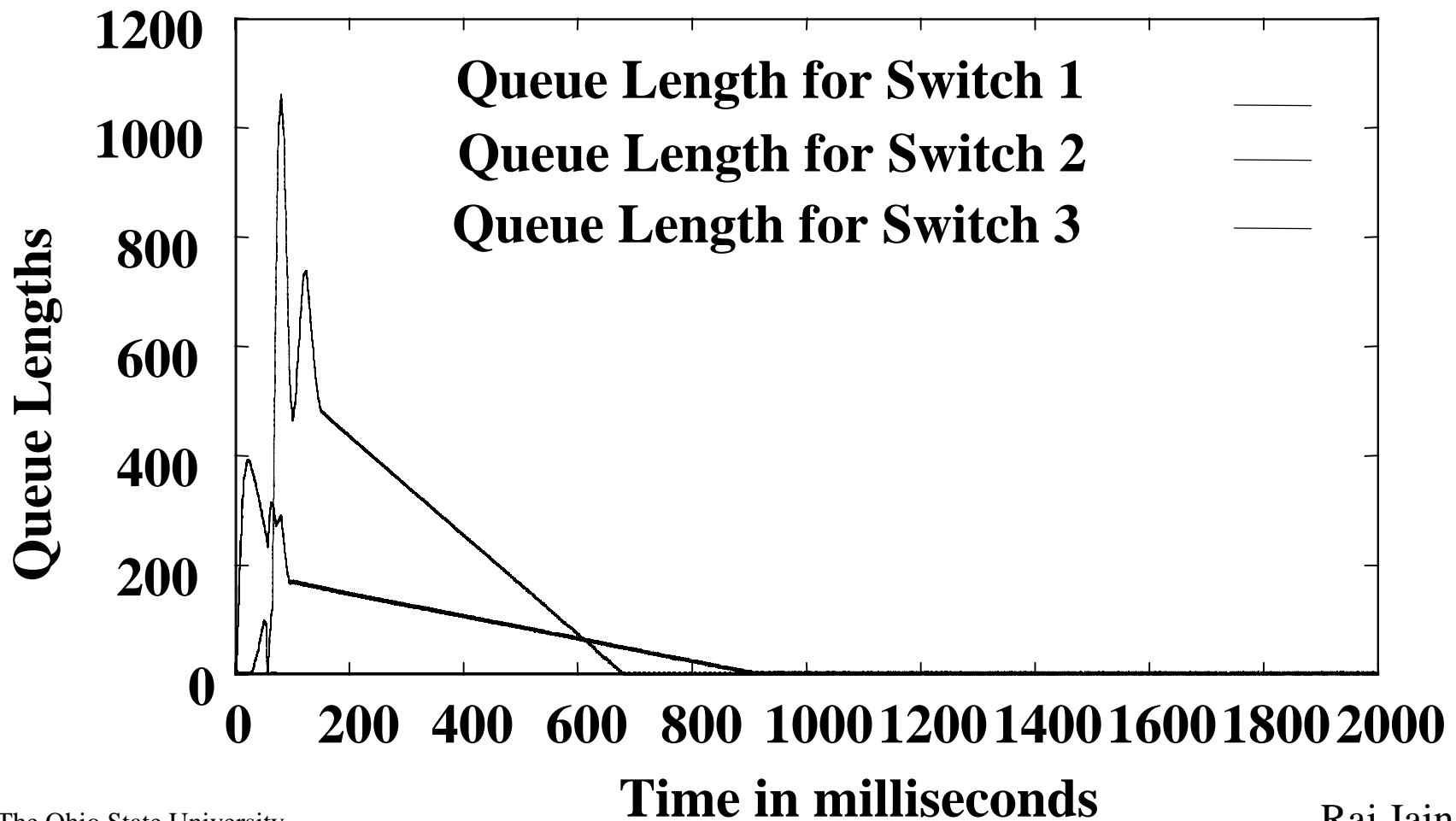
- Upstream Bottleneck, LINK3 = 5000 km, RIF = 1, interval = 5 ms

## WAN 4-leaf with upstream bottleneck: ACRs



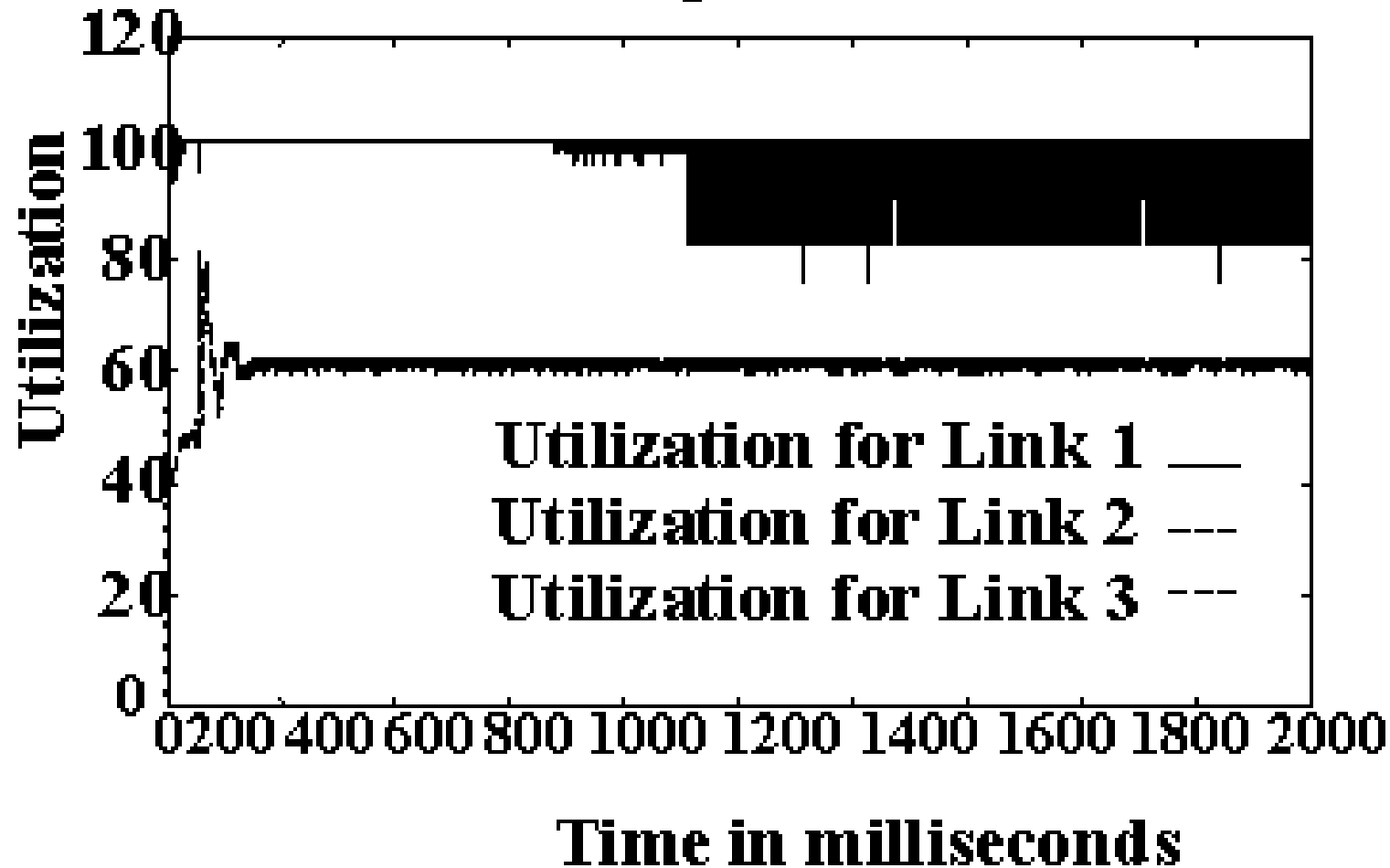
# Queue Lengths

WAN 4-leaf with upstream bottleneck



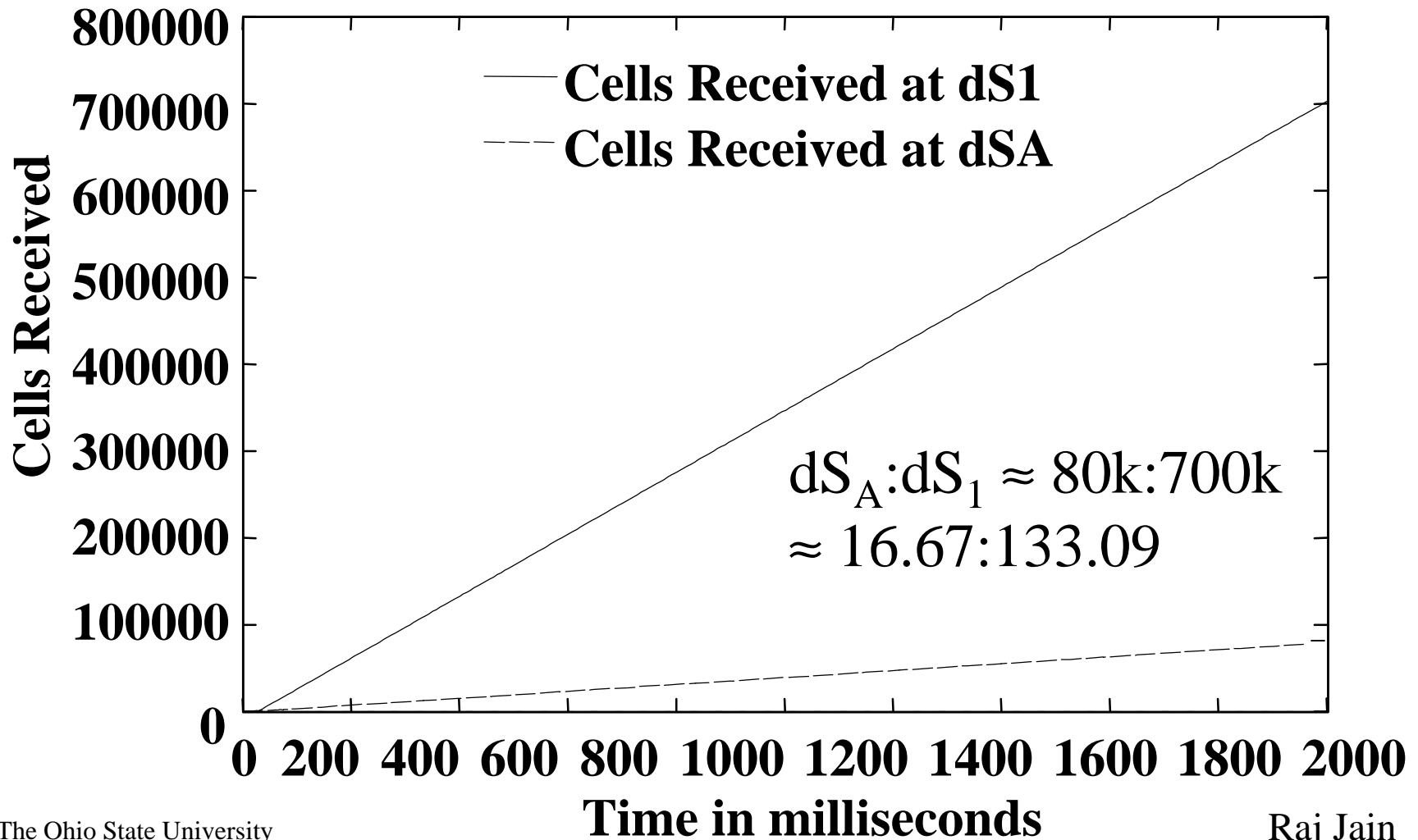
# Link Utilization

WAN 4-leaf with upstream bottleneck



# Cells Received

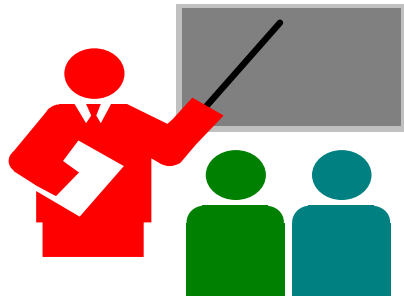
WAN 4-leaf with upstream bottleneck



# Lessons Learnt

- ❑ Avoid determining the effective number of active sources
- ❑ Avoid estimation of rates of sources, or determining if a source is bottlenecked at this link
- ❑ Use only aggregate measurements
- ❑ Do not use CCR values from BRM cells
- ❑ CCR from FRM cells can be used
- ❑ Using the maximum CCR in an interval, and exponentially averaging the maximum ER in the previous interval can improve performance
- ❑ Do not turn around RM cells at merging point

# Summary



- ❑ Multipoint-to-point ABR congestion control algorithms need to avoid problems that can arise in a naïve extension of point-to-point algorithms
- ❑ More extensive performance analysis is needed for proposed multipoint-to-point and multipoint-to-multipoint algorithms