

98-0154: Determining the Number of Active ABR Sources in Switch Algorithms

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

Department of CIS, The Ohio State University

Contact: jain@cis.ohio-state.edu

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



- q ERICA
- q New algorithm
- q Examples
- q Proof
- q Simulation Results

Original ERICA

End of measurement interval:

- q Target ABR Capacity
= Target Utilization \times Available Bandwidth
- q Load Factor $z = \text{ABR Input Rate} / \text{Target ABR Capacity}$
- q FairShare
= Target ABR Capacity / Number of Active VCs
- q VC's Share = Current Cell Rate / Load Factor z

BRM to be sent:

- q ER Calculated = Max (FairShare, VC's Share)

Number of Active VCs

- q FairShare
= Target Capacity/Number of Active VCs
- q Number of Active VCs: Number of VCs that sent one or more cells in the last ΔT interval
 \Rightarrow A VC that sends 1 cell is counted as an active VC
A VC that sends 1000 cells is also counted as an active VC
- q Activity of a VC is a discrete variable: 0 or 1

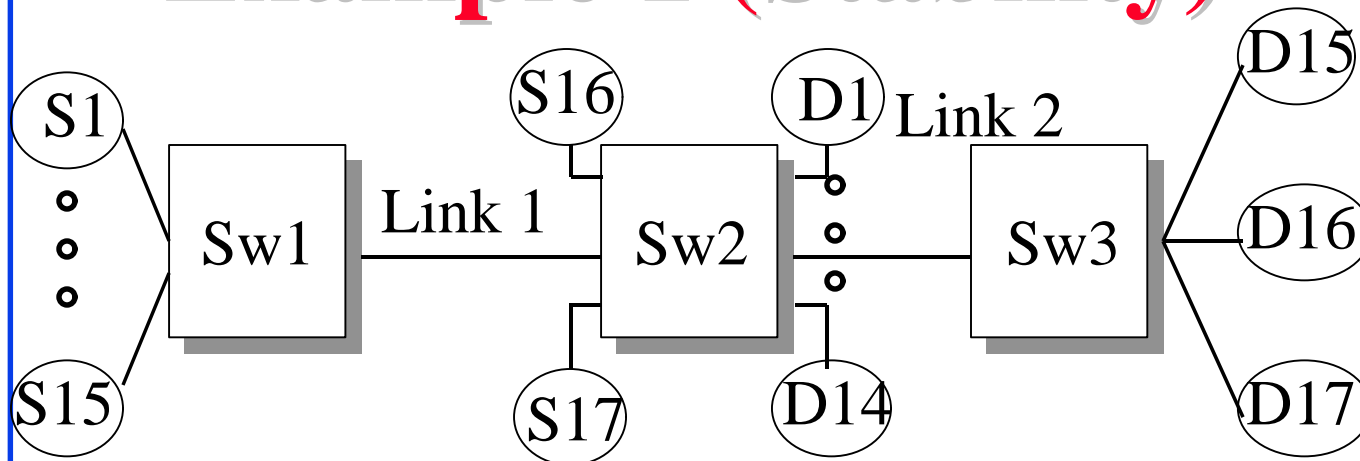
Effective Number of VCs

- q Idea: Activity can be a continuous variable.
⇒ A VC can have activity level anywhere between 0 and 1
- q Effective Number of VCs
= \sum_i Activity of i^{th} VC
- q FairShare = Target Capacity/Effective Number of VCs
- q Example: 3 sources with activity of 0.5, 0.75, 1
Available capacity = 149 Mbps
Target Utilization = 0.9
FairShare = $0.9 \times 149 / (0.5 + 0.75 + 1) = 59.6$ Mbps

Determining Activity Level

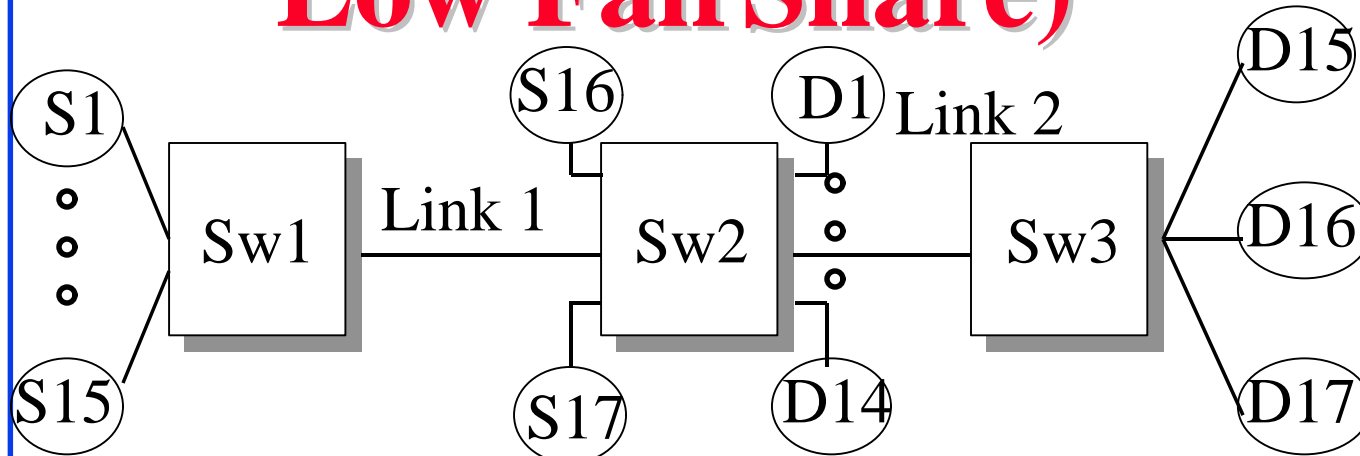
- q Activity level = $\text{Min}(1, \text{Source rate}/\text{FairShare})$
 \Rightarrow VCs operating $\geq \text{FairShare}$ are each counted as 1;
VCs operating $< \text{FairShare}$ only contribute a fraction
- q Effective number of VCs = $\sum_i \text{Activity level of VC } i$
- q FairShare =
Target ABR Capacity/Effective Number of VCs
- q Definitions are recursive
- q However, starting with any arbitrary value of FairShare, the procedure converges quickly

Example 1 (Stability)



- q Target capacity for Link 1 and Link 2 = 150 Mbps
- q For Sw2, $(S15, S16, S17) = (10, 70, 70)$
- q *Iteration 1*: FairShare = 70 Mbps
 - q Activity = $(10/70, 70/70, 70/70) = (1/7, 1, 1)$
 - q Effective # of VCs = $1 + 1 + 1/7 = 15/7$
- q *Iteration 2*: FairShare = Target capacity/Effective Number of VCs = $150/2.14 \approx 70$ Mbps

Example 2 (Rising from a Low FairShare)



q Rates = (10, 50, 90)

q Assume FairShare = 50

q *Iteration 1:*

q Activity = (10/50, 50/50, 1) = (0.2, 1, 1)

q Effective # of VCs = 0.2 + 1 + 1 = 2.2

q *Iteration 2:* FairShare = 150/2.2 \approx 70 Mbps

Example 3 (Dropping from a High FairShare)

- q Same configuration, rates = (10, 50, 90), FairShare = 75 Mbps
- q *Iteration 1:*
 - q Activity = $(10/75, 50/75, 1) = (0.13, 0.67, 1)$
 - q Effective # of VCs = $0.13 + 0.67 + 1 = 1.8$
- q *Iteration 2:* FairShare = $150/1.8 = 83$ Mbps
- q Assume sources send at new rates, except for S15
- q Activity = $(10/83, 83/83, 83/83) = (0.12, 1, 1)$
- q Effective # of VCs = $0.12 + 1 + 1 = 2.12$
- q FairShare = $150/2.12 \approx 70$ Mbps

Proof

- q *Claim:* This procedure leads to max-min fairness in all cases
- q *Proof:* Two Steps
 1. This is equivalent to MIT scheme
 2. MIT scheme leads to max-min fairness [Charny95]

Derivation of Step 1

q MIT Scheme: FairShare = [ABR Capacity
– $\sum_{i=1}^{N_u} R_{ui}$]/ N_o where:

R_{ui} = Rate of i^{th} underloading source ($1 \leq i \leq N_u$)

N_u = # of underloading VCs, N_o = # of overloading VCs

q FairShare * N_o = ABR Capacity – $\sum_{i=1}^{N_u} R_{ui}$

q FairShare * N_o + $\sum_{i=1}^{N_u} R_{ui}$ = ABR Capacity

q FairShare * [N_o + $\sum_{i=1}^{N_u} R_{ui}/\text{FairShare}$] = ABR
Capacity

q FairShare = ABR Capacity/ N_{eff} , where:

$N_{\text{eff}} = N_o + \sum_{i=1}^{N_u} R_{ui}/\text{FairShare}$

Benefits

- q Simulation results show that:
 - q Method works even with short measurement intervals and low rate sources
 - q Max-min fairness is achieved even without the previous fairness solution:

MaxAllocPrevious = maximum allocation
in the previous interval, initialized to FairShare

IF (load factor $z > 1 + \delta$)

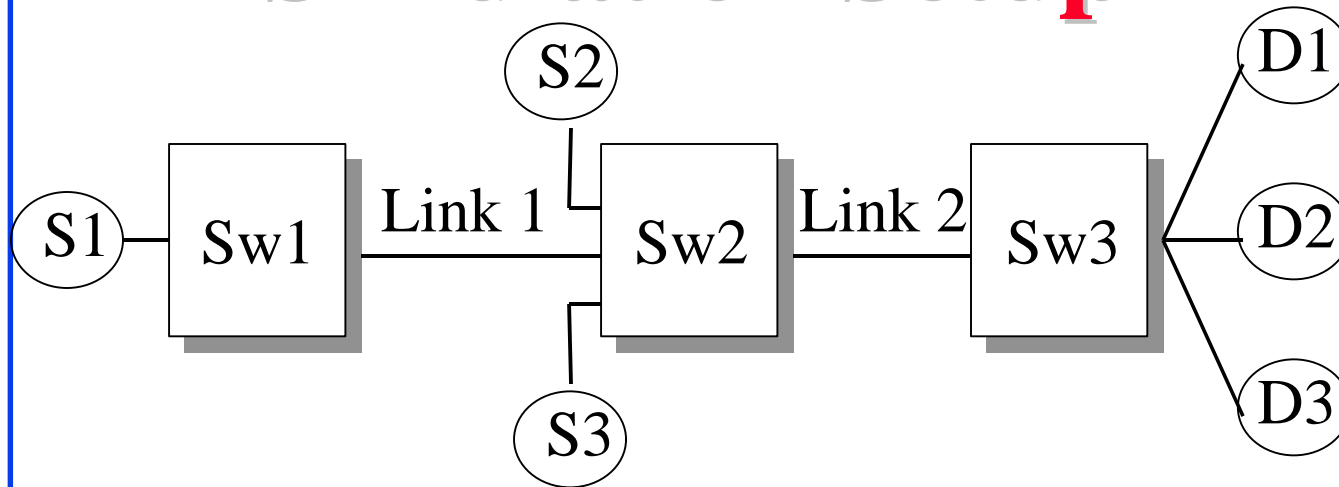
THEN

ER = Max (CCR/z, FairShare)

ELSE

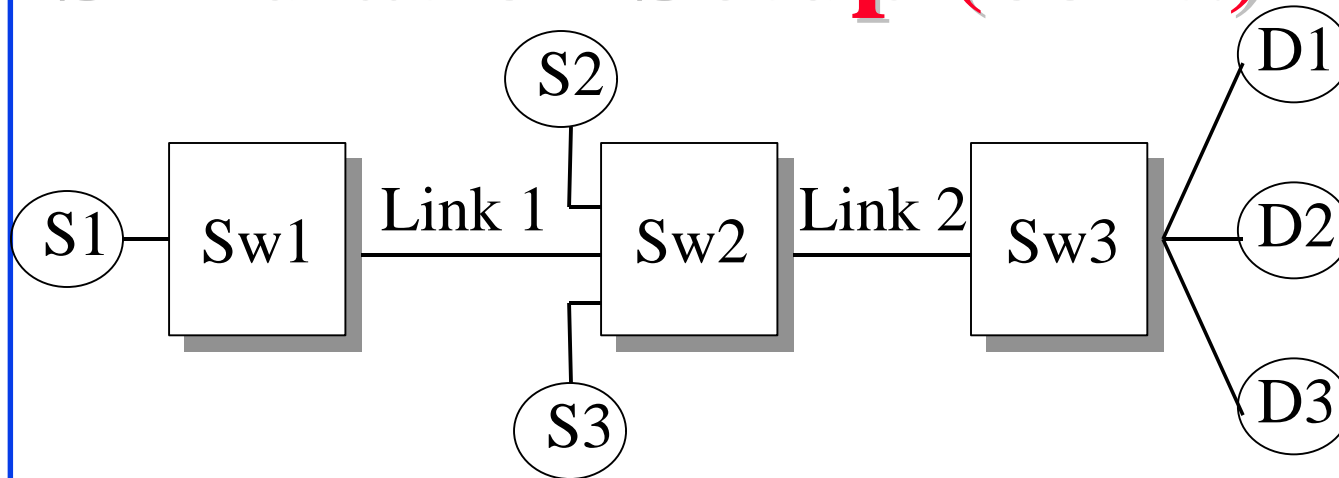
ER = Max (CCR/z, *MaxAllocPrevious*)

Simulation Setup



- q \forall links: bandwidth = 155.52 Mbps, length = 1000 km
- q All VCs are bidirectional
- q S1 is bottlenecked at 10 Mbps, ICR for S2 = 30 Mbps, for S3 = 110 Mbps, $S1+S2+S3=150$ Mbps
- q Tests if S2 and S3 reach same ACR, using bandwidth left over by S1

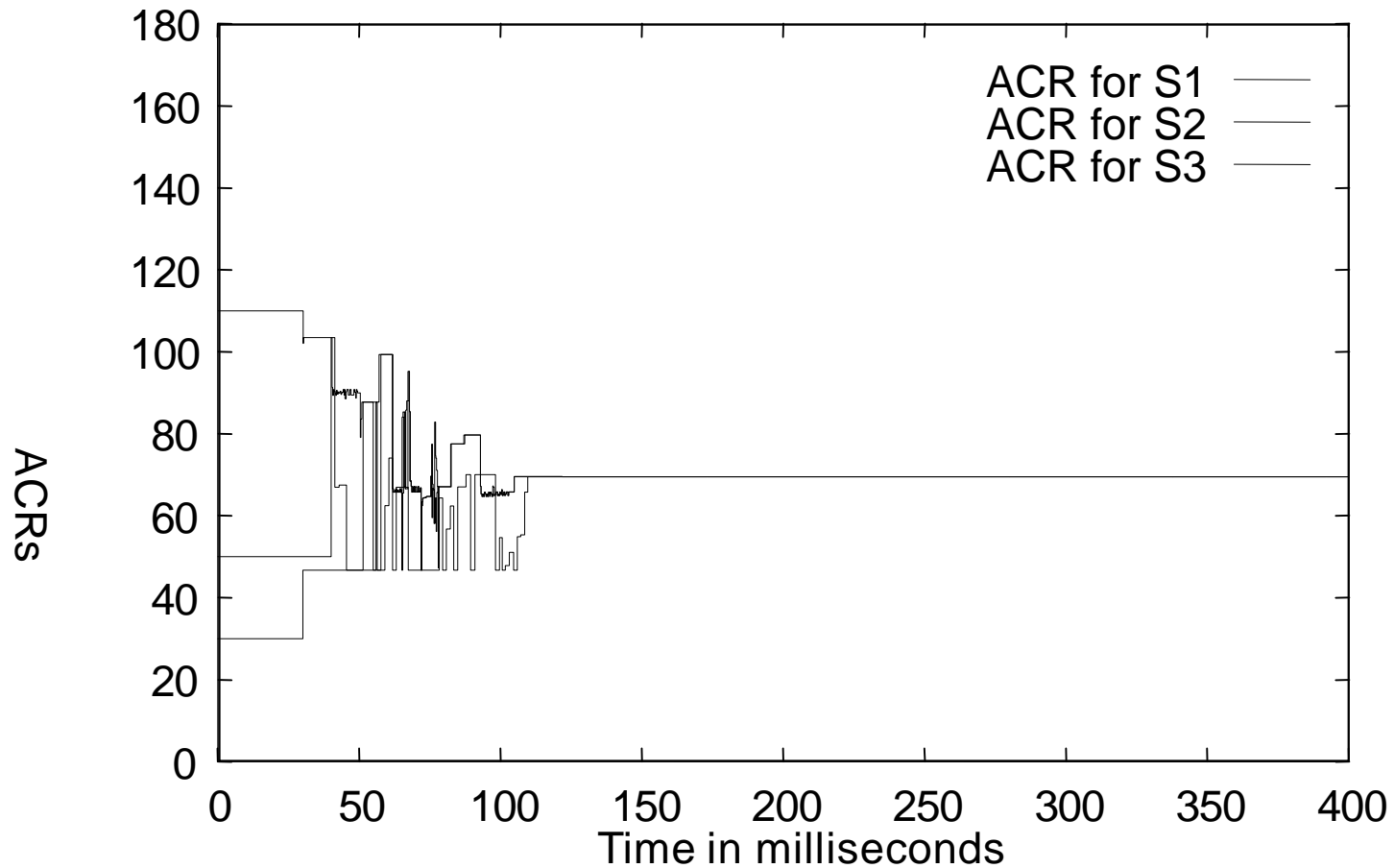
Simulation Setup (cont.)



- q Same as configuration used in examples, except that S1 VC is bottlenecked at *S1 itself* (not Link 1), to show effect of source bottlenecks
- q RIF = 1, TBE = large
- q Switch target utilization parameter = 90%
- q Switch interval = min (time (100 cells), 1 ms)

Results: ERICA

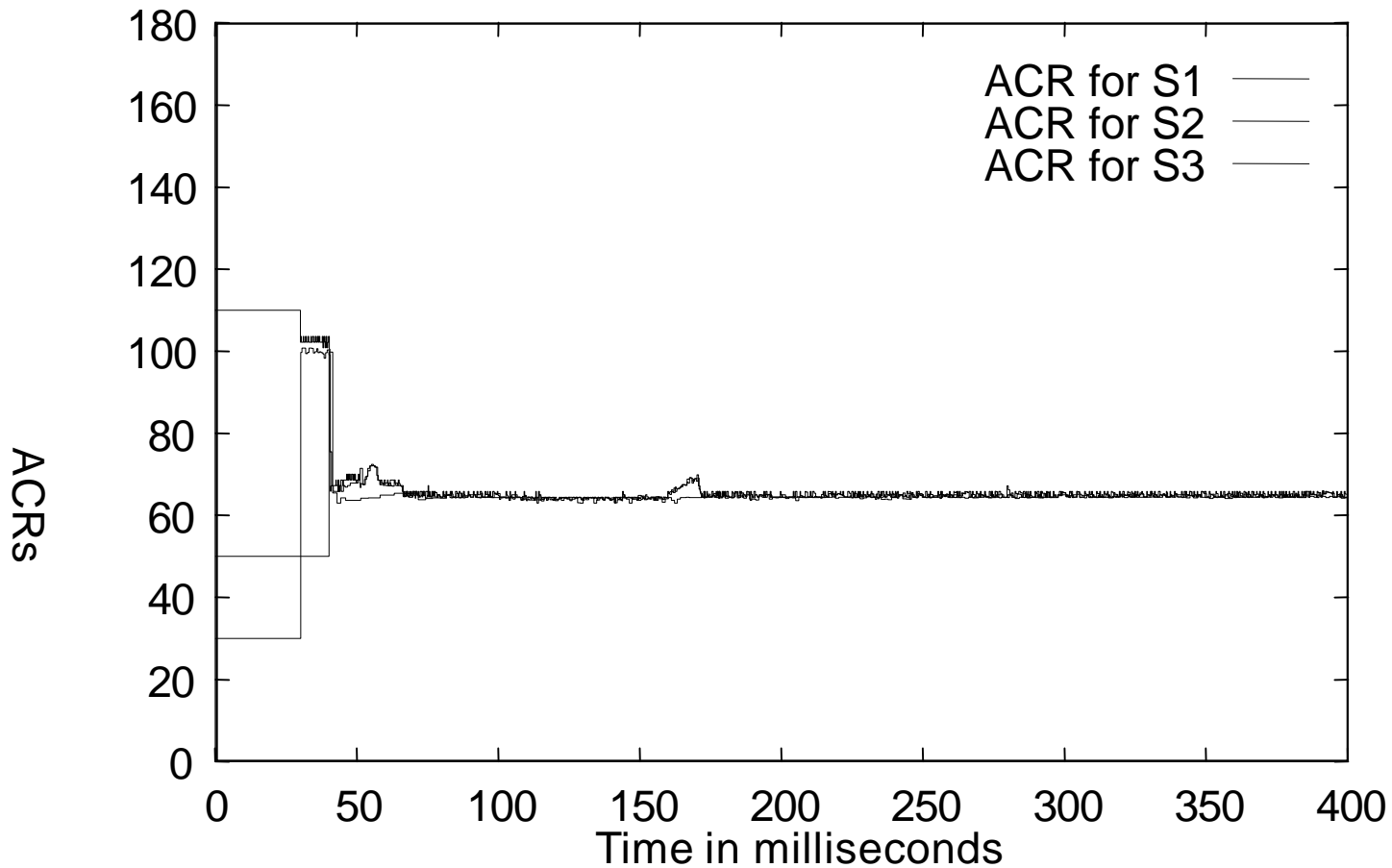
WAN Bottlenecked: ACRs



ERICA with *MaxAllocPrevious* solution attains fairness

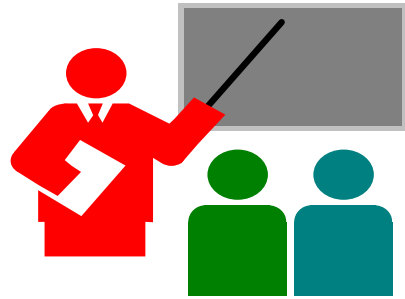
Results: New Method

WAN Bottlenecked: ACRs



New method also attains fairness. Note faster convergence

Conclusions



- q New method distinguishes underloading and overloading connections to compute activity levels, effective # of active connections, and fair share.
- q Method is provably max-min fair, and maintains the *fast* transient response, queuing delay control, and simplicity of ERICA. It overcomes the need for the ERICA fairness steps and is less sensitive to measurement interval length.