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

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

Raj Jain is now at
Washington University in Saint Louis
Jain@cse.wustl.edu
http://www.cse.wustl.edu/~jain/
Overview

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

End of measurement interval:

- Target ABR Capacity
  \[= \text{Target Utilization} \times \text{Available Bandwidth}\]

- Load Factor \( z = \frac{\text{ABR Input Rate}}{\text{Target ABR Capacity}} \)

- FairShare
  \[= \frac{\text{Target ABR Capacity}}{\text{Number of Active VCs}}\]

- VC’s Share
  \[= \frac{\text{Current Cell Rate}}{\text{Load Factor } z}\]

BRM to be sent:

- ER Calculated \(=\) Max (FairShare, VC’s Share)
Number of Active VCs

- FairShare
  \[ \text{FairShare} = \frac{\text{Target Capacity}}{\text{Number of Active VCs}} \]

- Number of Active VCs: Number of VCs that sent one or more cells in the last \( \Delta T \) interval
  \[ \Rightarrow \text{A VC that sends 1 cell is counted as an active VC} \]
  \[ \text{A VC that sends 1000 cells is also counted as an active VC} \]

- Activity of a VC is a discrete variable: 0 or 1
Effective Number of VCs

- Idea: Activity can be a continuous variable. 
  \[ \Rightarrow \text{A VC can have activity level anywhere between 0 and 1} \]

- Effective Number of VCs
  \[ = \sum_i \text{Activity of } i^{\text{th}} \text{ VC} \]

- FairShare = Target Capacity/Effective Number of VCs

- Example: 3 sources with activity of 0.5, 0.75, 1
  - Available capacity = 149 Mbps
  - Target Utilization = 0.9
  - FairShare = \( 0.9 \times 149 / (0.5 + 0.75 + 1) = 59.6 \text{ Mbps} \)
Determining Activity Level

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

- Target capacity for Link 1 and Link 2 = 150 Mbps
- For Sw2, (S15, S16, S17) = (10, 70, 70)
- **Iteration 1**: FairShare = 70 Mbps
  - Activity = (10/70, 70/70, 70/70) = (1/7, 1, 1)
  - Effective # of VCs = 1 + 1 + 1/7 = 15/7
- **Iteration 2**: FairShare = Target capacity/Effective Number of VCs = 150/2.14 ≈ 70 Mbps
Example 2 (Rising from a Low FairShare)

- Rates = (10, 50, 90)
- Assume FairShare = 50
- **Iteration 1:**
  - Activity = (10/50, 50/50, 1) = (0.2, 1, 1)
  - Effective # of VCs = 0.2 + 1 + 1 = 2.2
- **Iteration 2:** FairShare = 150/2.2 ≈ 70 Mbps
Example 3 (Dropping from a High FairShare)

- Same configuration, rates = (10, 50, 90), FairShare = 75 Mbps

- **Iteration 1:**
  - Activity = (10/75, 50/75, 1) = (0.13, 0.67, 1)
  - Effective # of VCs = 0.13 + 0.67 + 1 = 1.8

- **Iteration 2:** FairShare = 150/1.8 = 83 Mbps

- Assume sources send at new rates, except for S15

- Activity = (10/83, 83/83, 83/83) = (0.12, 1, 1)

- Effective # of VCs = 0.12 + 1 + 1 = 2.12

- FairShare = 150/2.12 ≈ 70 Mbps
Proof

- **Claim:** This procedure leads to max-min fairness in all cases

- **Proof:** Two Steps
  1. This is equivalent to MIT scheme
  2. MIT scheme leads to max-min fairness [Charny95]
MIT Scheme: FairShare = [ABR Capacity

\[ - \sum_{i=1}^{\text{Nu}} R_{ui} \]/N_o \]

where:

- \( R_{ui} \) = Rate of \( i^{th} \) underloading source \( (1 \leq i \leq \text{Nu}) \)
- \( \text{Nu} \) = # of underloading VCs, \( N_o \) = # of overloading VCs

\[ \text{FairShare} \times N_o = \text{ABR Capacity} - \sum_{i=1}^{\text{Nu}} R_{ui} \]

\[ \text{FairShare} \times N_o + \sum_{i=1}^{\text{Nu}} R_{ui} = \text{ABR Capacity} \]

\[ \text{FairShare} \times [N_o + \sum_{i=1}^{\text{Nu}} R_{ui}/\text{FairShare}] = \text{ABR Capacity} \]

\[ \text{FairShare} = \text{ABR Capacity}/N_{\text{eff}}, \text{ where:} \]

\[ N_{\text{eff}} = N_o + \sum_{i=1}^{\text{Nu}} R_{ui}/\text{FairShare} \]
Benefits

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

\[
\text{MaxAllocPrevious} = \text{maximum allocation in the previous interval, initialized to FairShare}
\]

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

THEN

\[
ER = \text{Max (CCR}/z, \text{FairShare)}
\]

ELSE

\[
ER = \text{Max (CCR}/z, \text{MaxAllocPrevious)}
\]
Simulation Setup

- ∀ links: bandwidth = 155.52 Mbps, length = 1000 km
- All VCs are bidirectional
- S1 is bottlenecked at 10 Mbps, ICR for S2 = 30 Mbps, for S3 = 110 Mbps, S1+S2+S3=150 Mbps
- Tests if S2 and S3 reach same ACR, using bandwidth left over by S1
Same as configuration used in examples, except that S1 VC is bottlenecked at *S1 itself* (not Link 1), to show effect of source bottlenecks

- RIF = 1, TBE = large
- Switch target utilization parameter = 90%
- Switch interval = \( \text{min (time (100 cells), 1 ms)} \)
Results: ERICA

ERICA with *MaxAllocPrevious* solution attains fairness
Results: New Method

New method also attains fairness. Note faster convergence.
Conclusions

- New method distinguishes underloading and overloading connections to compute activity levels, effective # of active connections, and fair share.
- 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.