# 97-0832: Fairness for ABR Multipoint-to-Point Connections

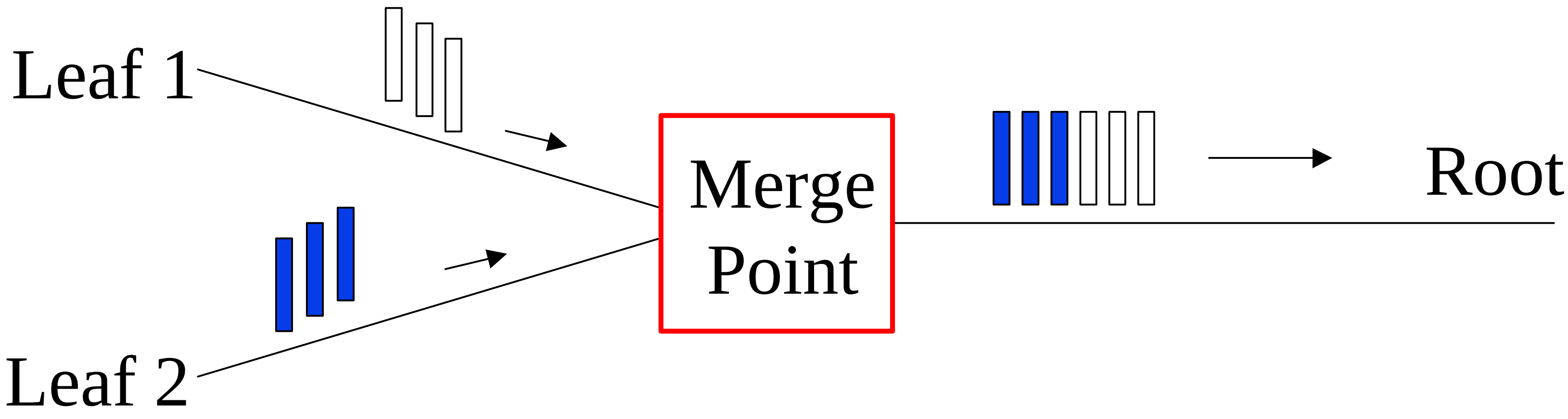
**Sonia Fahmy, Raj Jain, Rohit Goyal,**
**and Bobby Vandalore**
Department of CIS, The Ohio State University
**Sastri Kota**, Lockheed Martin Telecommunications
**Pradeep Samudra,** Samsung Telecom America, Inc.

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


- Multipoint-to-point VCs
- Cell Interleaving Solutions
- Multipoint-to-point Algorithms
- Fairness Definitions, Examples and Comparisons
- Design and Implementation Issues

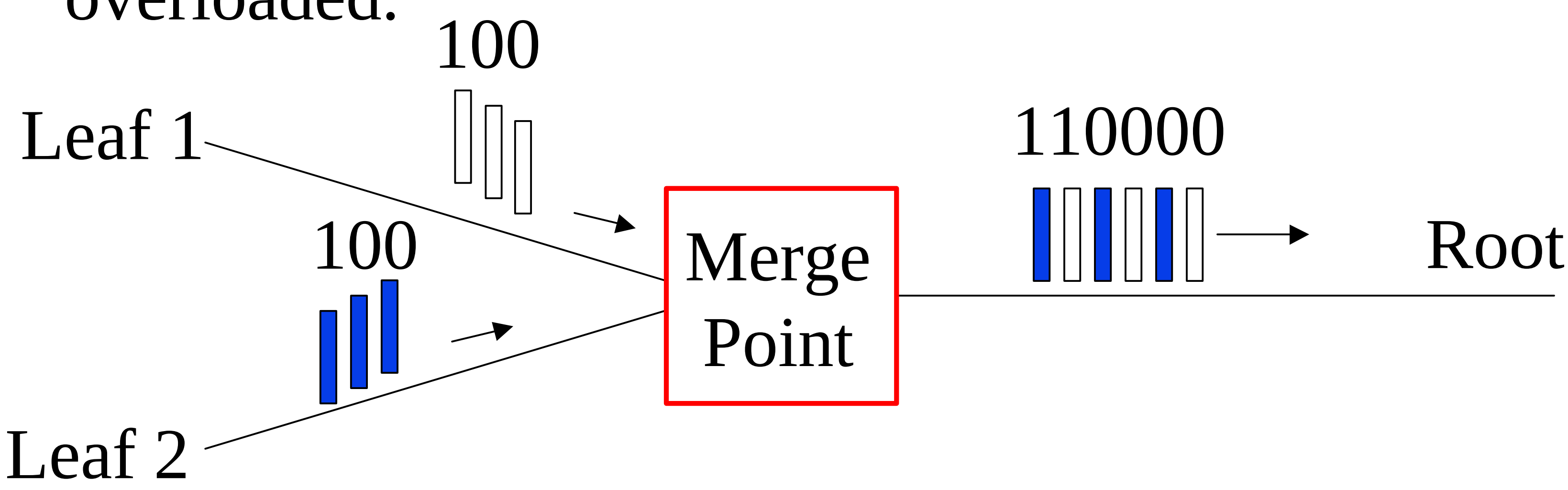
# Multipoint-to-Point VCs

- A multipoint-to-point VC can have more than one concurrent sender
- Traffic at root = Σ Traffic originating from leaves
- How can bandwidth be allocated fairly?



# Cell Interleaving Solutions

- AAL 3/4: Limited sources, high overhead, unused.
- VC Mesh: Each source sets up a 1-to-n multicast VC. Not scalable.
- Multicast Server (MCS): Senders send to MCS, which forwards data on a 1-to-n VC. Can become overloaded.

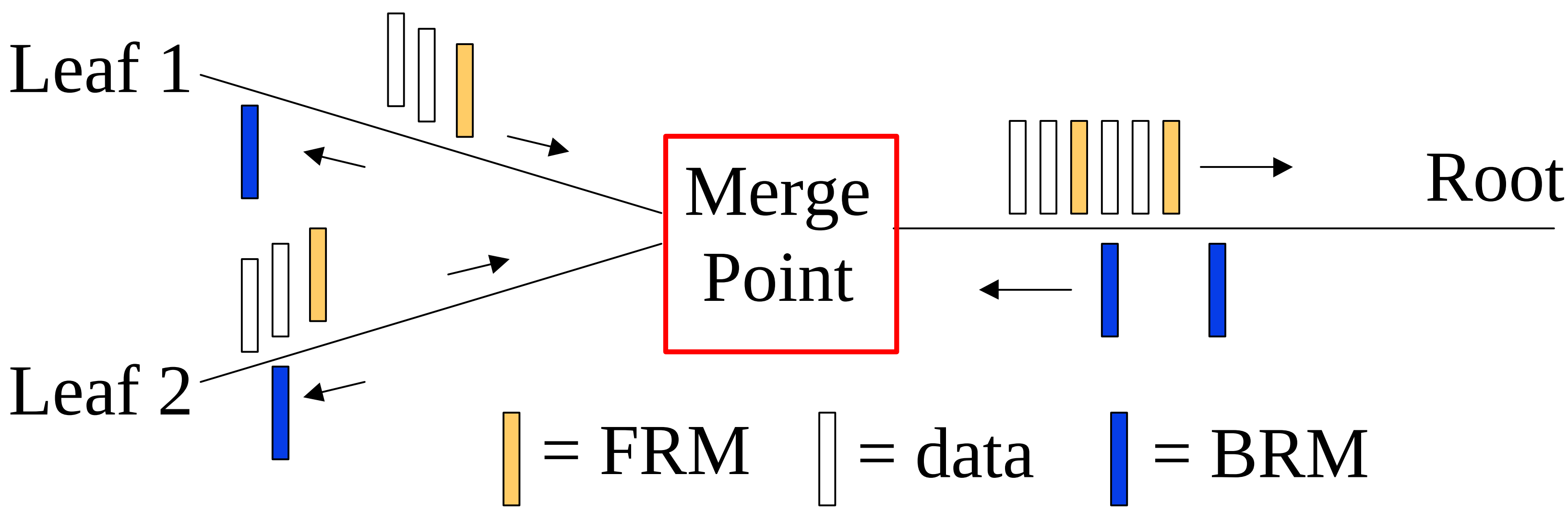


# Cell Interleaving (Cont)

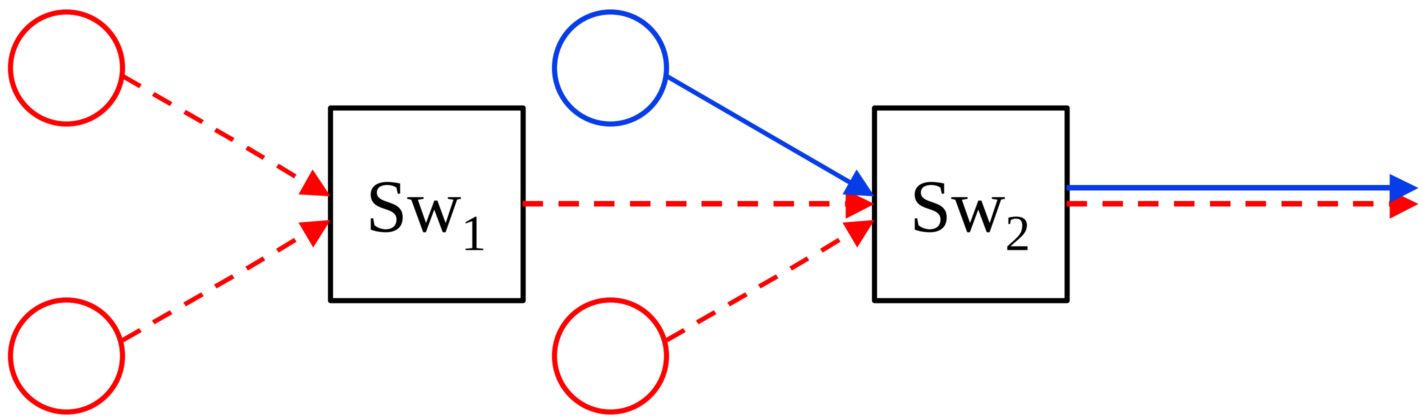
- Tokens: Only token holder can transmit.
  High overhead and delay.
- VP merge: VCI = sender ID
  VPs are used for other purposes.
- VC merge: Buffer at merge point till EOM bit = 1.
  Requires memory and adds to traffic burstiness and latency.
- Sub-channel multiplexing: Use GFC.
  May not scale well.

# Multipoint-to-Point Algorithms

- Maintain a bit at the merge point for each flow being merged. Bit = 1 $\Rightarrow$ 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.



# Sources, VCs, and Flows



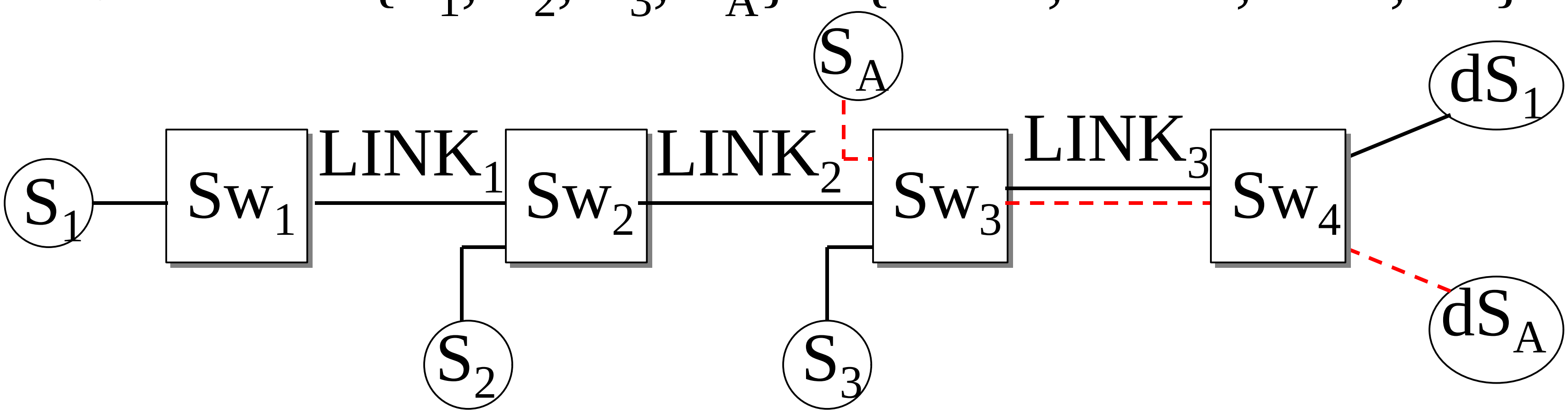

- $Sw_2$ has to deal with
  - Two VCs: Red and Blue
  - Four sources: Three red sources and one blue source
  - Three flows: Two red flows and one blue

# Fairness Definitions

- Source-based: N-to-one connection = N one-to-one connections $\Rightarrow$ Use max-min fairness among sources
- VC/Source-based:
  1. Allocate bandwidth fairly among VCs
  2. For each VC, allocate fairly among its sources
- Flow-based: Flow = VC coming on an input link. Switch can easily distinguish flows.
- VC/Flow-based:
  1. Allocate bandwidth fairly among VCs
  2. For each VC, allocate fairly among its flows

# Example I

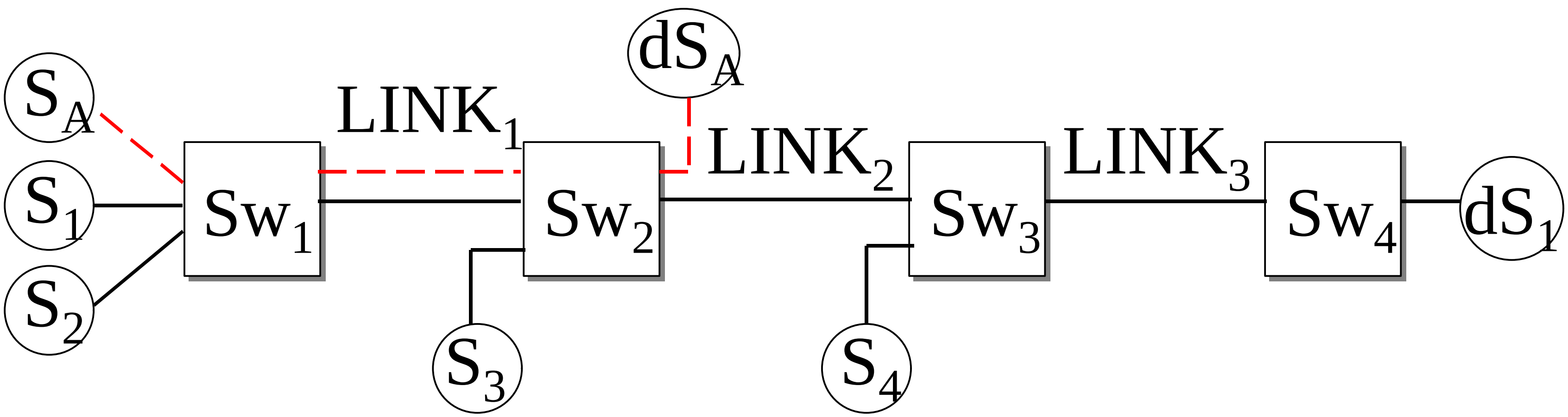
- How is the bandwidth of $LINK_3$ allocated?
- Source: $\{S_1, S_2, S_3, S_A\} \leftarrow \{37.5, 37.5, 37.5, 37.5\}$
- VC/Source: $\{S_1, S_2, S_3, S_A\} \leftarrow \{25, 25, 25, 75\}$
- Flow: $\{S_1, S_2, S_3, S_A\} \leftarrow \{25, 25, 50, 50\}$
- VC/Flow: $\{S_1, S_2, S_3, S_A\} \leftarrow \{18.75, 18.75, 37.5, 75\}$




All links are 150 Mbps

# Example II

- How is left-over capacity on $LINK_3$ allocated?
- Source:$\{S_1,S_2,S_3,S_4,S_A\} \leftarrow \{16.7,16.7,58.3,58.3,16.7\}$
- VC/Src:$\{S_1,S_2,S_3,S_4,S_A\} \leftarrow \{12.5,12.5,62.5,62.5, 25\}$
- Flow:$\{S_1,S_2,S_3,S_4,S_A\} \leftarrow \{16.7,16.7,41.7,75,16.7\}$
- VC/Flow:$\{S_1,S_2,S_3,S_4,S_A\} \leftarrow \{12.5,12.5,50,75,25\}$




All links are 150 Mbps, except $LINK_1$ which is 50 Mbps

# Comparison

- Source-based versus VC/source-based: In source-based, a multipoint-to-point VC with N concurrent senders is allocated N/K times the bandwidth allocated to a VC with K concurrent senders (if all senders are bottlenecked on the same link).
  Is pricing based on senders or on VCs?
- Flow-based and VC/flow-based: Suffer from a "beat-down"-like problem. Sources whose flow crosses a larger number of merge points are allocated less bandwidth. But this may be acceptable in practical situations.

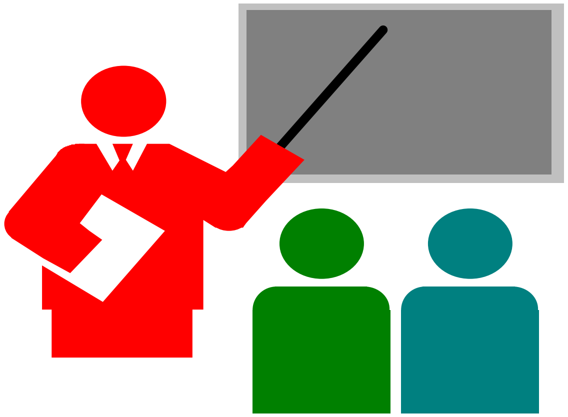
# Design Issues

- Per-source/VC/flow accounting are equivalent for point-to-point, but different for multipoint-to-point.
  - Avoid per-source accounting (with VC merge).
  - Aggregate flow values for per-VC accounting.
- Per-source accounting is possible with VP merge.
- Using downstream rate allocations is necessary for all types except source-based (see next slide).
- Do destinations or merge points generate BRMs?
- For scalability, overhead and delays should not increase with the increase of the levels of the tree.

# Implementations

- Source: Simplest to implement.
  - Avoid any per-source accounting, and estimation of rates or number of active sources.
- VC/Source: Bi-level operation, i.e., compute VC allocations, and source allocations.
  - Use downstream allocations since VC bandwidth needs to be divided among VC sources.
- Flow: Separate flows are merged into one flow $\Rightarrow$ Must use downstream allocations.
- VC/Flow: Bi-level operation to estimate both (VC and flow) allocations based upon load and capacity.

# Summary



- VP merge versus VC merge
- Fairness based on sources, VCs, or flows
- Use of per-source/VC/flow accounting
- Multipoint ABR algorithms can offer tradeoffs between complexity, noise, transient response, overhead and scalability

# Motion 1

- Add section 2 of 97-0832 to the baseline text of living list item 97-001

# Motion 2

- Add section 7 of 97-0832 to the baseline text of living list item 97-001