Source Flow Control computation & simulation

Lihao Chen (lihao.chen@huawei.com)
Lily Lv (lvyunping@huawei.com)

HUAWEI

SFC use cases before and now

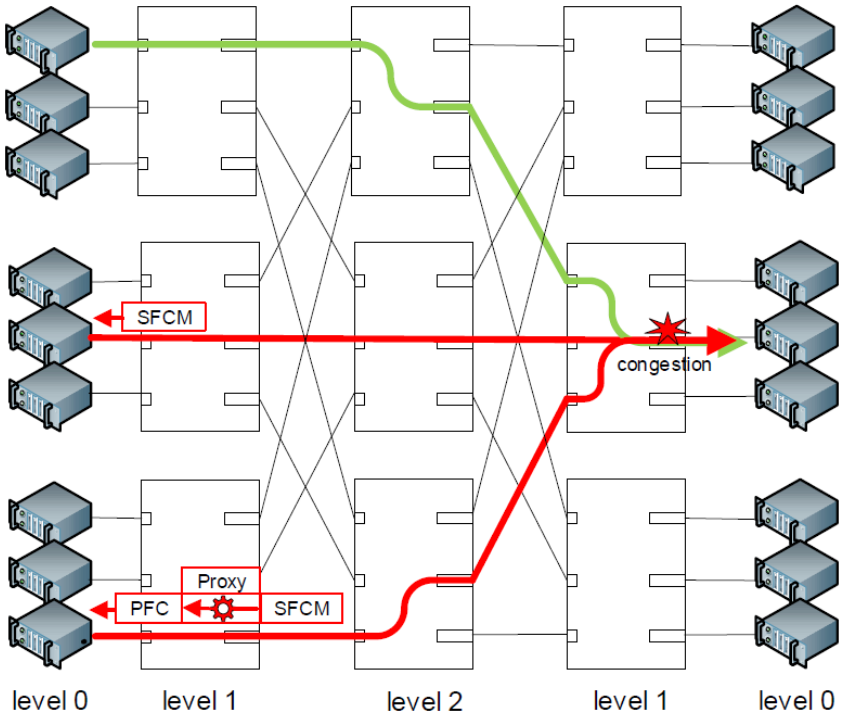
- All previous discussions in Nendica (2021) and 802.1Qdw (before Feb. 2023) are about using Source Flow Control or remote PFC in high-performance DCNs.
  - The need and benefits of SFC have been presented, including simulations and theoretical calculations.

- Discussions since July 2024 are about using SFC in AIDCNs, data center networks designed for AI computing, e.g., AI model training and inferences.
  - This presentation also focuses on SFC for AIDCN, providing use cases, computations and simulations.

- *"Note: From the author's current perspective, it is coincidentally the case that SFC can address both scenarios and can be implemented using a single protocol design."*

Topology

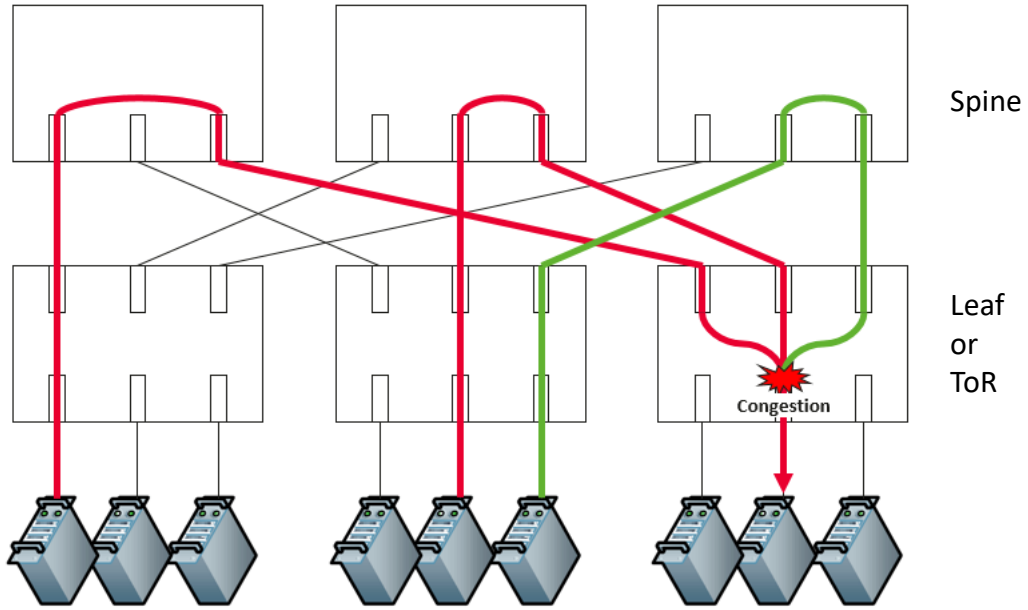
Previously used



fold

unfold

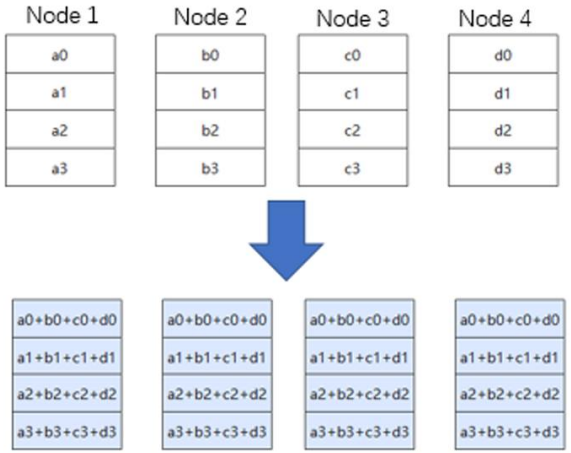
2-layer CLOS

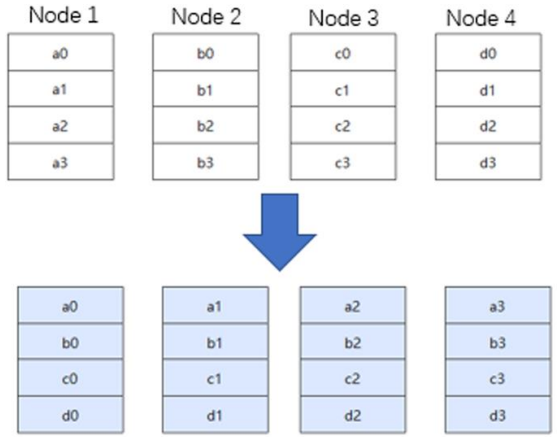


HUAWEI

Incast congestion

- Collective communications with huge bandwidth demands during AI training and inference cause N-to-1 incast, especially Allreduce and Alltoall.





https://mentor.ieee.org/802.1/dcn/24/1-24-0028-04-ICne-aicn-report-draft.pdf

In both scenarios, node 1 needs to receive data ranging from hundreds of MB to hundreds of GB from Node 2, 3, and 4 at the same time!

  - Allreduce is commonly used in TP (Tensor Parallelism) & DP (Data Parallelism).
    - Typical use case: NCCL (NVIDIA Collective Communication Library).
  - Alltoall is commonly used in EP (Expert Parallelism) for MoE (Mixture of Experts) architecture.
    - Typical use case: DeepEP (communication library designed by DeepSeek).
- Packet-spray load balancing is commonly used in AIDCNs. Network-based solutions require reordering at the edge bridge next to the receiving end-station. A high degree of out-of-order packets can lead to incast congestion at the edge bridge.
- 'Slow receiver' sends PFC to the edge bridge.

Why SFC?

- The reaction time is extremely important!
  - The DCQCN algorithm generates congestion notification at most every 50 usec and the fastest it reduce the rate is by half, therefore in the very best case it takes two steps to reduce the rate to 1/4.

	PFC	DCQCN	SFC
Reaction Time (Time To Stop RP Egress)	Slowest >50-200 usec	Middle > 100 usec	Fastest Between 2-10 usec
Time To Stop CP Ingress	Fastest 2 usec	Slowest > 100 usec	Middle-Fast Between 2-15 usec
Reaction Point Location	Per Class	Per Flow	Per Class / Per Interface / Per Flow
Handles Short Lived Flows	Yes	No	Yes
Handles Long Lived Flows	No	Yes	Yes
Micro-Burst Control	Yes	Poor	Yes
Congestion Spread Free	No	Yes	Yes
Deadlock Free	No	Yes	Yes

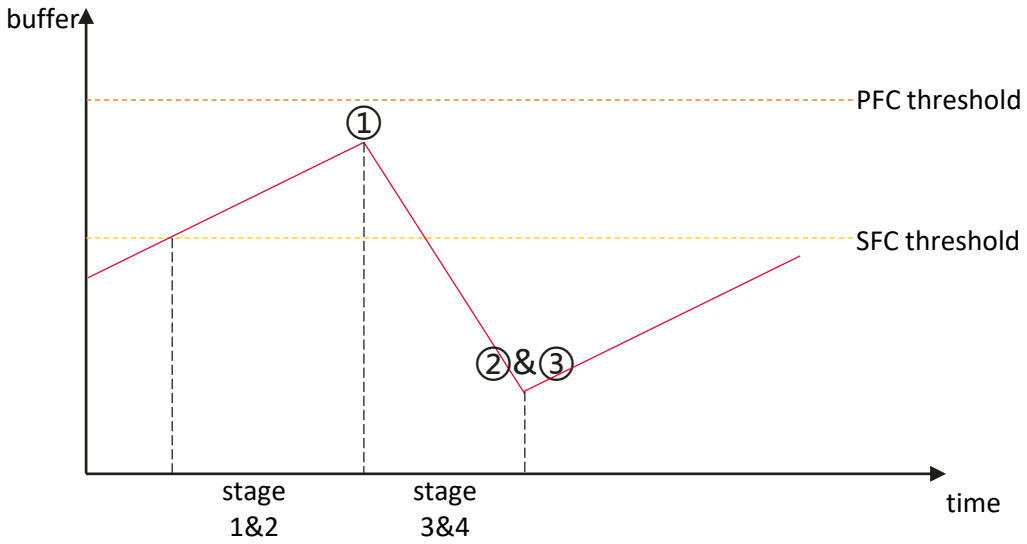
https://www.ieee802.org/1/files/public/docs2022/new-bottorff-sfc-0322-v6.pdf

  - While the bandwidth and the switch overall forwarding capacity have a Moore's-Law-like growth, the buffer capacity does not.
- Different use case.
  - DCQCN was not explicitly designed to address the on-off traffic patterns of AI collective communication. E.g., need to avoid improper speed up during the OFF stage.

SFC Computation

- Congestion point's buffer usage changes after SFC triggered:

Stage	Description	Duration	Congestion point buffer
1	SFC threshold reached and SFCM sent.	t_1 + ½ RTT	Increasing.
2	Sender PAUSE, packets still on-the-fly.	t_2 + ½ RTT	Increasing.
3	On-the-fly drains.	t_{PAUSE} - ½ RTT	Decreasing.
4	Sender resume.	½ RTT	Decreasing.
5	New packets arrive.	/	Increasing.



- SFC reacts on N-to-1 incast, **not causing**
  - ①Buffer usage exceeding the PFC threshold. (SFC reaction too slow!)

$$Threshold_{PFC} - Threshold_{SFC} \geq Buffer_{increase} = (\alpha N - 1)Bw \times (RTT + t_1 + t_2)$$

  - ②SFC threshold still exceeded after the Pause. (SFC under-reacted! Pause time too short.)

$$Buffer_{increase} - Buffer_{decrease} = (\alpha N - 1)Bw \times (RTT + t_1 + t_2) - Bw \times t_{PAUSE} \leq 0$$

  - ③Under-utilization of bandwidth. (SFC over-reacted! Pause time too long.)

$$Threshold_{SFC} + Buffer_{increase} - Buffer_{decrease} = Threshold_{SFC} + (\alpha N - 1)Bw \times (RTT + t_1 + t_2) - Bw \times t_{PAUSE} \geq 0$$

  - $Threshold\ and\ Buffer\ in\ bit,\ N{:}N\ to\ 1\ incast,\ \frac{1}{N} \leq \alpha \leq 1,\ Bw{:}Bandwidth,\ t_{PAUSE}{:}Pause\ time\ in\ SFCM,\ RTT{:}round\ trip\ time$
  - $t_1{:}time\ needed\ to\ detect\ the\ congestion\ and\ to\ construct\ and\ send\ the\ SFCM$
  - $t_2{:}time\ needed\ from\ receiving\ the\ SFCM\ to\ stop\ sending\ (and\ potentially\ including\ the\ translation\ on\ proxy\ bridge)$

HUAWEI

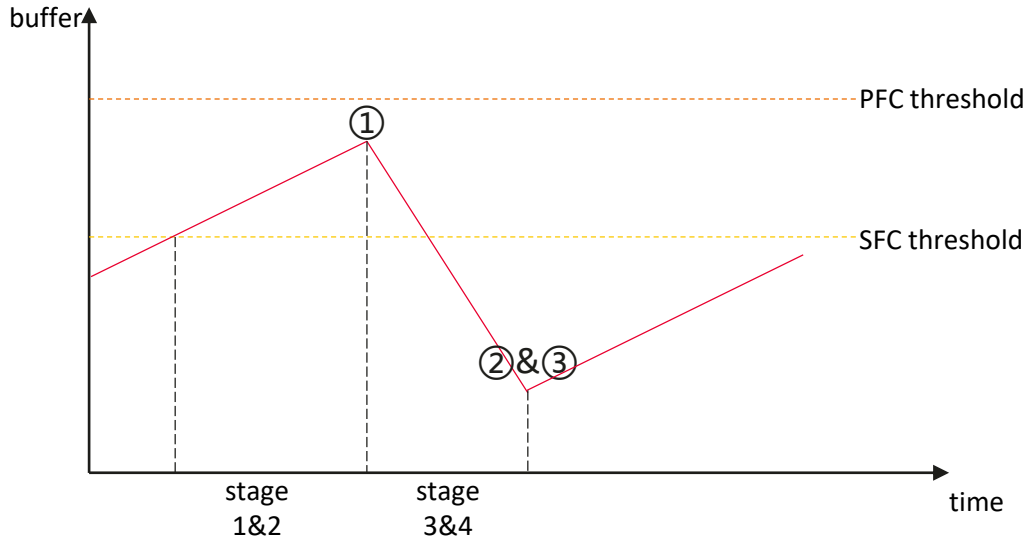
SFC Parameter Calculation Example

$$Threshold_{PFC} - Threshold_{SFC} \geq Buffer_{increase} = (\alpha N - 1)Bw \times (RTT + t_1 + t_2)$$

$$Buffer_{increase} - Buffer_{decrease} = (\alpha N - 1)Bw \times (RTT + t_1 + t_2) - Bw \times t_{PAUSE} \leq 0$$

$$Threshold_{SFC} + Buffer_{increase} - Buffer_{decrease} = Threshold_{SFC} + (\alpha N - 1)Bw \times (RTT + t_1 + t_2) - Bw \times t_{PAUSE} \geq 0$$

- Substitute the data: Bw=400Gbps, N=7, RTT=20us, t_1=t_2=10us, α=0.18:
  - $Threshold_{PFC} - Threshold_{SFC} \geq 520KByte$
  - $t_{PAUSE} \geq 10.4us$
- Set $Threshold_{SFC}$=1MByte: or
  - $t_{PAUSE} \leq 30.4us$
- Set t_{PAUSE}=30us:
  - $Threshold_{SFC} \geq 980KByte$

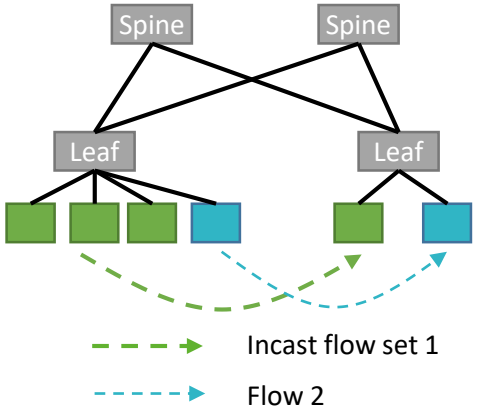


- This calculation process could be given as informative in the Annex to help users setting PFC and SFC triggering threshold and SFC pause time based on switch buffer capacities, round trip time as well as other influencing factors.

HUAWEI

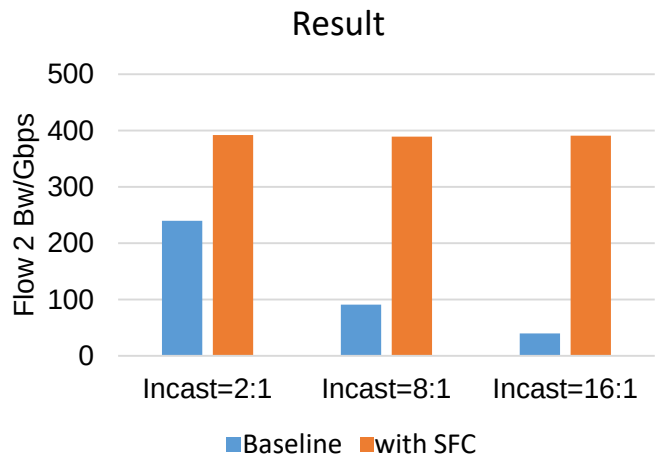
SFC Simulation

Incast baseline simulation

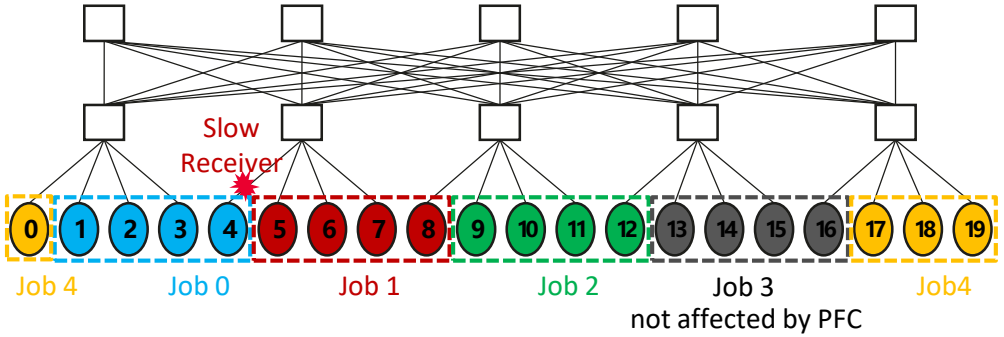


Settings: Flow set 1 with different degrees of incast, see how flow 2 is influenced. Packet-based load-balancing.

Environment: ESL simulation platform, with the end-stations simulate the NIC's behavior of NVIDIA GPUs.

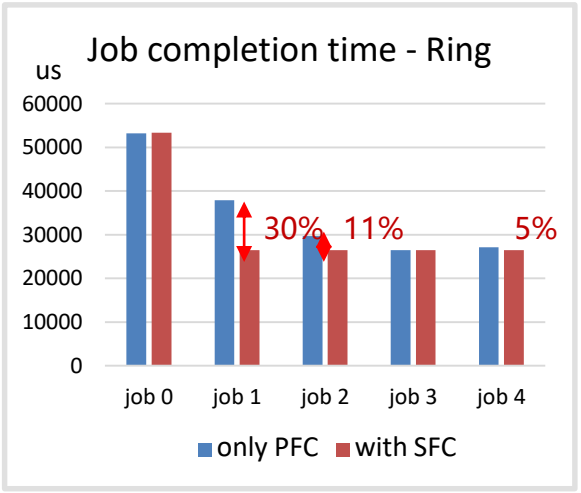


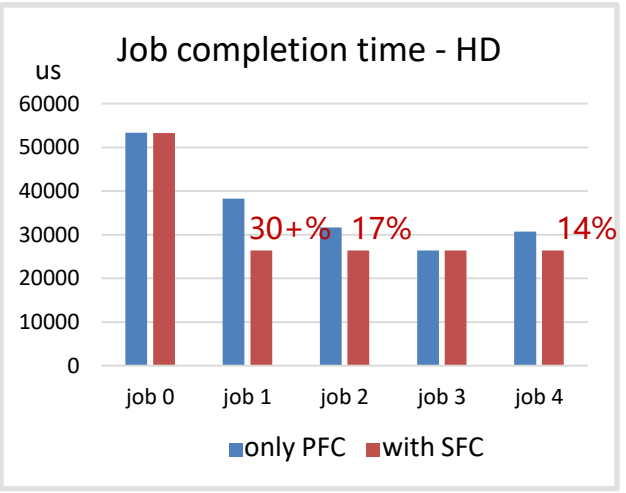
Collective communication with slow-receiver simulation



Settings:

1. each link 100Gbps
2. Allreduce data: 200MByte
3. Allreduce algorithm: Ring/HD
4. Slow Receiver: No.4, at 50% speed

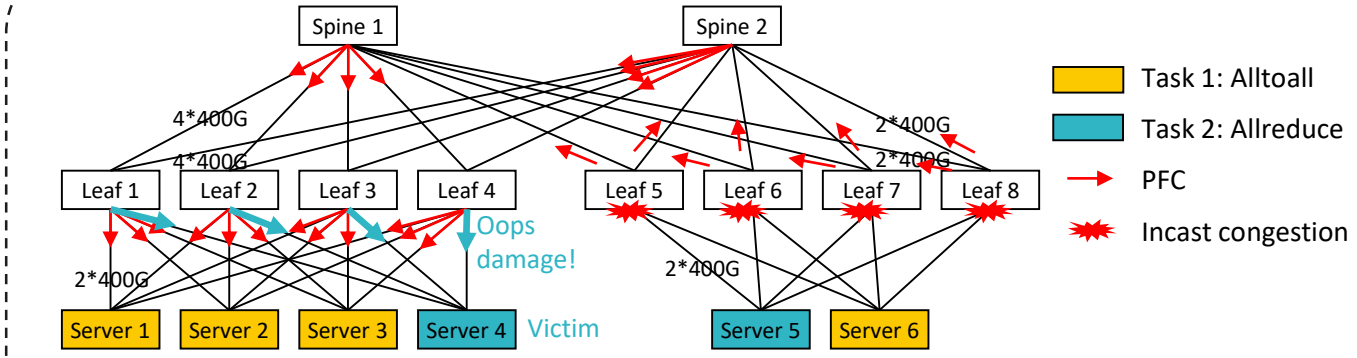




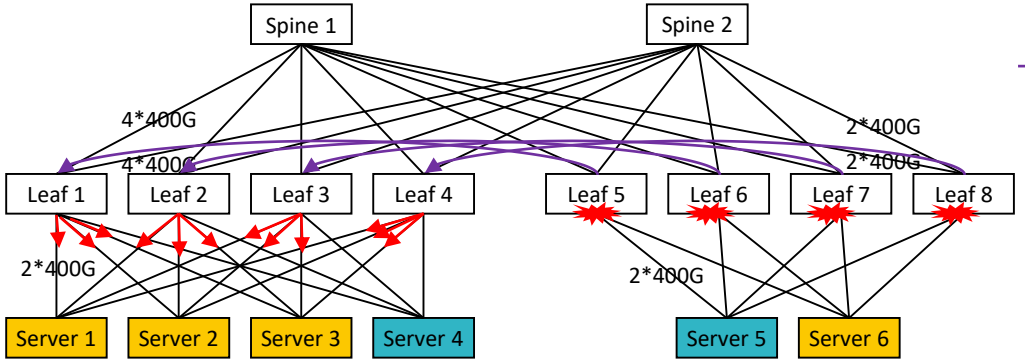
HUAWEI

SFC verification

Verification environment settings

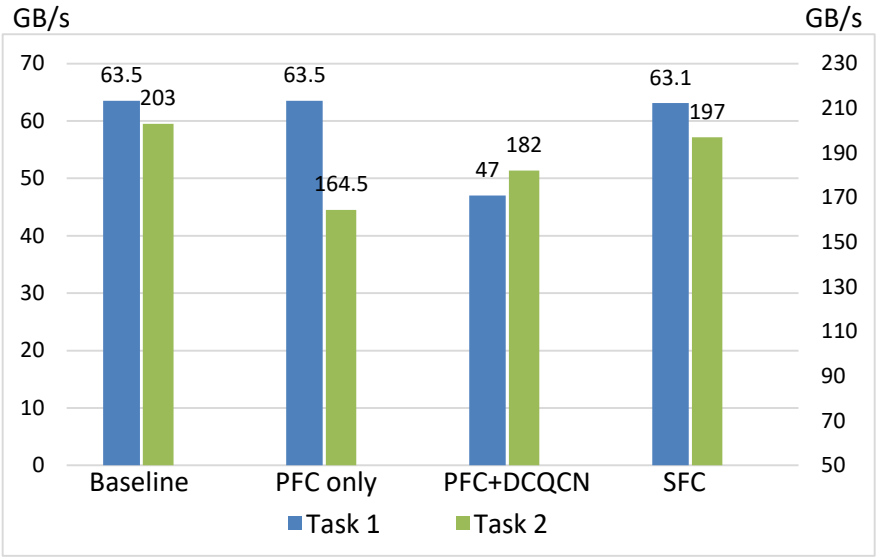


Task 1 Alltoall (sender: server 1-3, receiver: server 6) causes congestion on Leaf switch 5-8, PFC oops damages Task 2.



Using SFC (proxy-mode), PFCs are sent to server 1-3, not server 4, avoiding unnecessary PFC spread and oops damage.

Results



Case 1: The baseline, i.e., run task 1 and task 2 separately.

Case 2: PFC only. Task 2 is oops damaged because of PFC spread.

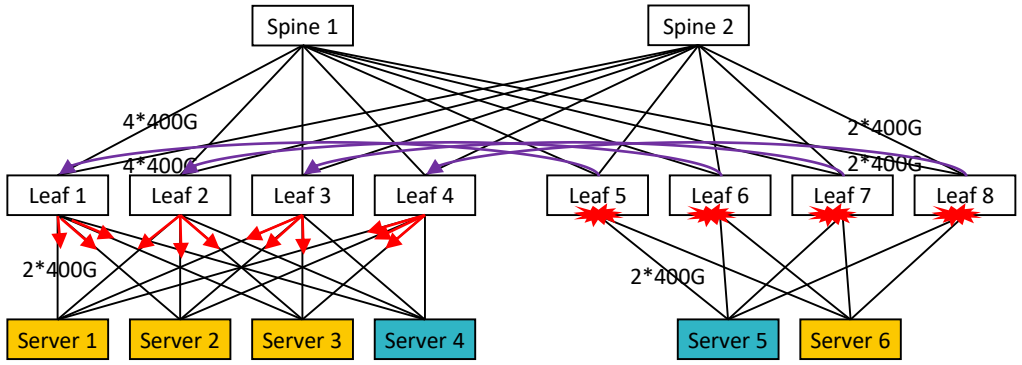
Case 3: Task 1 deteriorated because of DCQCN's relatively slow reaction and inaccurate control for on-off flows.

Case 4: Good test 2 performance without significant influence on task 1.

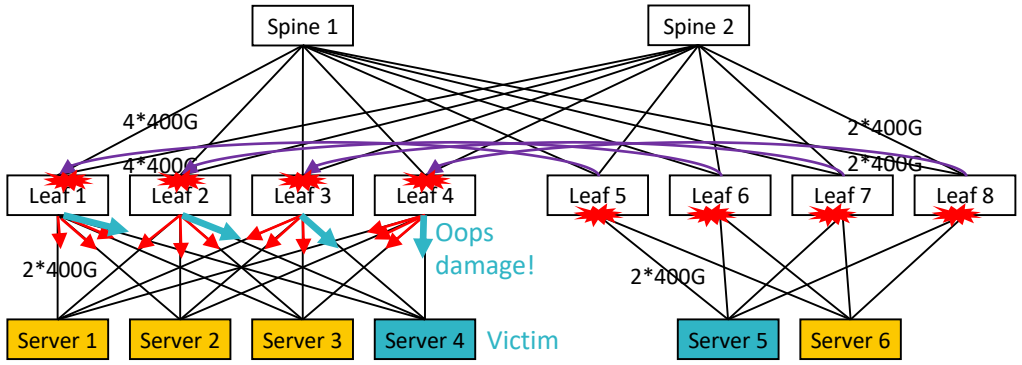
HUAWEI

Another potential approach

- “The edge bridge intercepts the SFCM and participates in the protocol by stopping or slowing transmission into the network. Then PFC from the edge bridge to the connected system happens (or doesn't happen) when the edge bridge buffering fills.” Let’s call it **bSFC** (bridge-reacted SFC).
- And let’s call the other one (either the end-station receives SFCM or the proxy-bridge translates SFCM to PFC to the end-station) **eSFC** (end-station-reacted SFC).
- The author suggests that the key distinction between bSFC and eSFC lies in bSFC's prioritization of leveraging edge bridge buffers to ‘absorb congestion’ caused by incast bursts, and only triggering PFC when buffer capacity is exceeded. However, this approach may not be ideal for AI workloads due to their extreme bandwidth demands, where buffer capacity is relatively limited and often shared across ports. Aggressively occupying buffers could introduce unpredictable instability in overall network performance. Additionally, it also risks triggering unnecessary PFC pauses (oops damage).



eSFC



bSFC

HUAWEI

Suggestions

- Move forward with the eSFC.
  - Contributions, including computational analysis and simulations, have validated the rationality and effectiveness of the eSFC approach for both high-performance DCNs and AIDCNs.

- The calculation process in this contribution could be given as informative in the Annex to help users setting SFC parameters based on their use cases, e.g. buffer capacities, RTT as well as other influencing factors.

- SFC should be provided as a tool, just like PFC and QCN.
  - The author's motivation is that SFC demonstrates particularly significant and unique benefits in AIDCN scenarios.
  - Based on a review of prior contributions, the author cautiously suggests that both scenarios (AIDCN & high-performance DCN) can share a unified SFC mechanism and protocol design.
  - Even if future work reveals the need to transmit entirely distinct parameters, this could be accommodated through Type-Length-Value (TLV) fields in the SFCM design.