Towards Fine-grained and Practical Flow Control for Datacenter Networks

Wenxue Li, Chaoliang Zeng, Jinbin Hu, Kai Chen

(iSING Lab @ HKUST)

Background

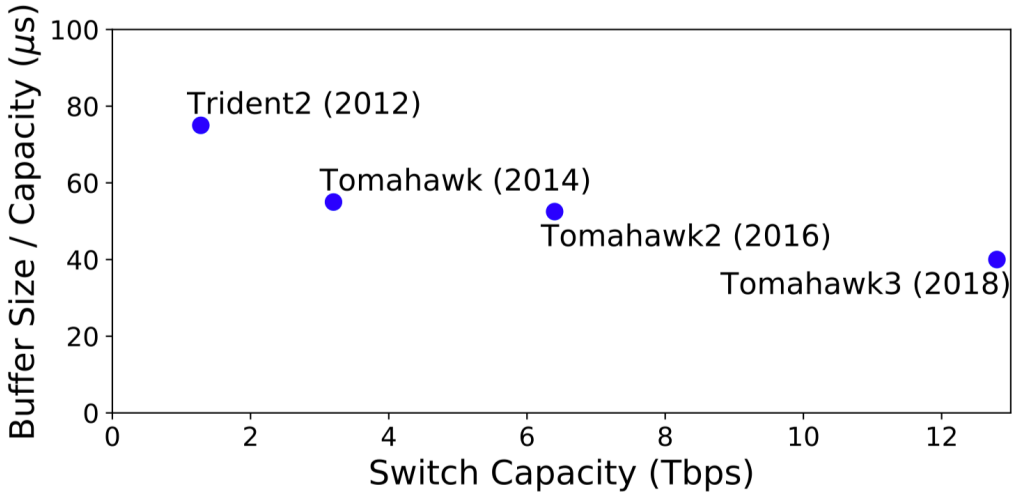
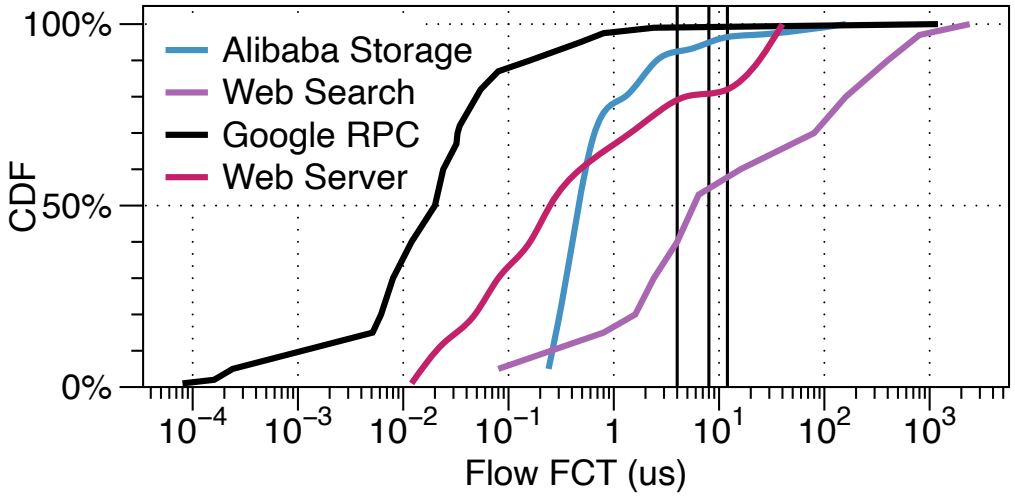


Image by the courtesy of BFC (Goyal etc., NSDI 2022)

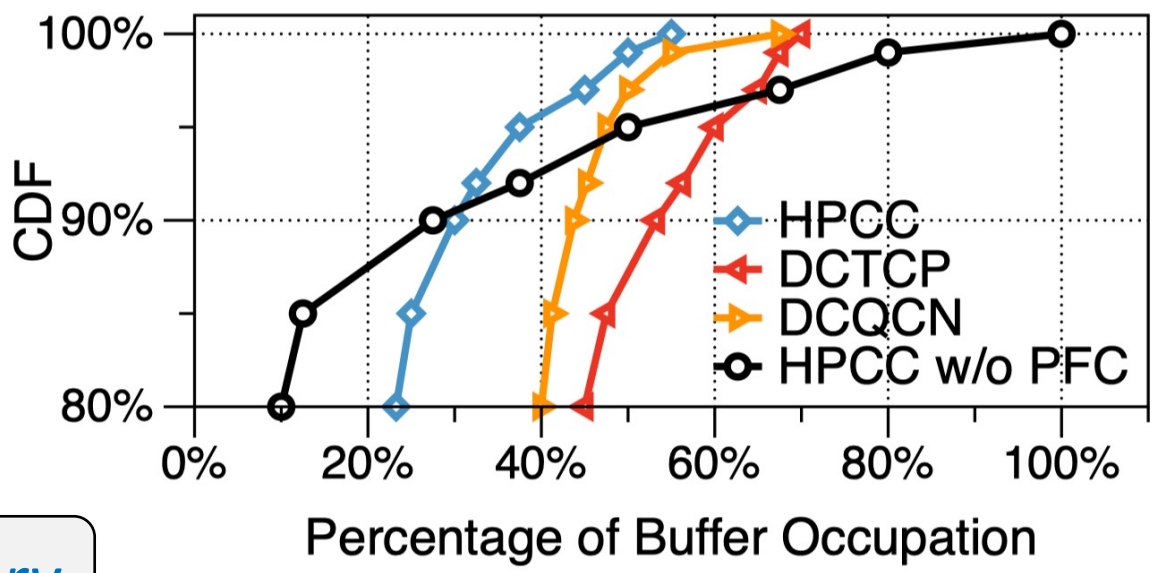


- Trend: Rapidly increasing link speed (switch capacity).
- However, switch buffer size lags behind switch capacity.

- Rising link speeds result in **increasingly short flows** ⇒ potentially induce greater burstiness in network traffic.

Insufficiency of End-to-End CC

- End-to-end CCs face challenges: senders need at least one RTT to receive the receiver-echoed signals ➔ **a loss of control over short flows.**
- **Observation** (via experiment):
  - ➢ CC alone experiences high tail buffer occupation.
  - ➢ CC + PFC reduces the buffer occupation.



Per-hop flow control (FC) is necessary.

Existing Flow Control Schemes are Insufficient

- Per-hop FC controls upstream entity within a 1-Hop RTT.

  1-Hop RTT (1∼2 µs) <<< end-to-end RTT (tens of µs)

- However, ☹️
  - ➢ PFC is coarse-grained ➔ Deadlock, Head-of-line blocking, etc.
  - ➢ Ideal FC allocates a dedicated queue to every flow ➔ fine-grained but impractical.
  - ➢ SOTA FC scheme, BFC, demands too many queues and compromises isolation granularity when queues are limited.

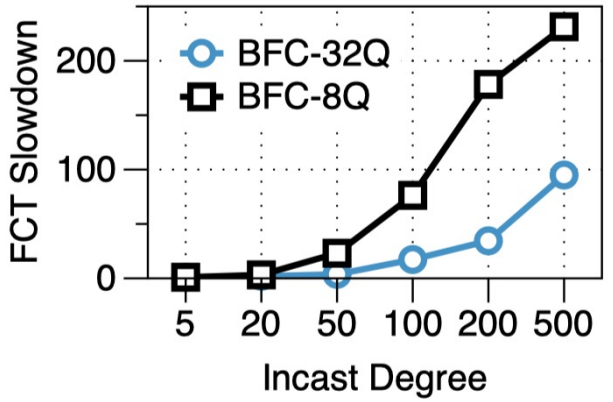
BFC compromises isolation granularity

- BFC dynamically assigns a dedicated queue to each active flow.
- However, when queues are limited, BFC permits multiple flows to share a queue and *manages all flows within the same queue collectively*

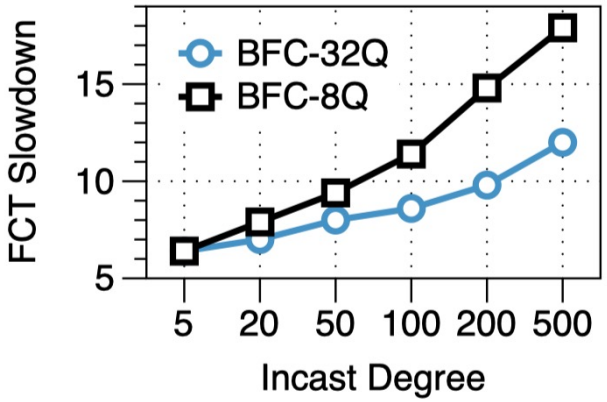
  ⇒ **its performance critically depends on the # of available queues.**

The # of available queues is limited in practice. ☹

(details in the paper)



(a) Tail for short flows



(b) Avg for large flows

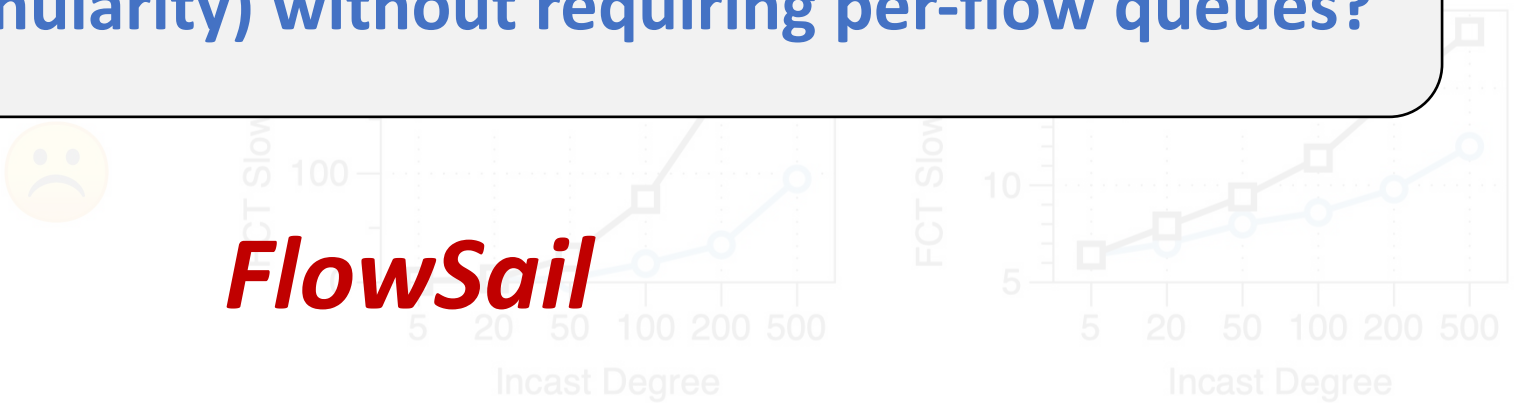
Our Goal

- BFC dynamically assigns a dedicated queue to each active flow.
- When queues are limited, BFC permits multiple flows to share a queue and *manages all flows within the same queue collectively*.
- ☹ available queues is limited in practice.

(details in the paper)

Can we design an FC scheme that offers fine-grained control (i.e., per-flow granularity) without requiring per-flow queues?

FlowSail



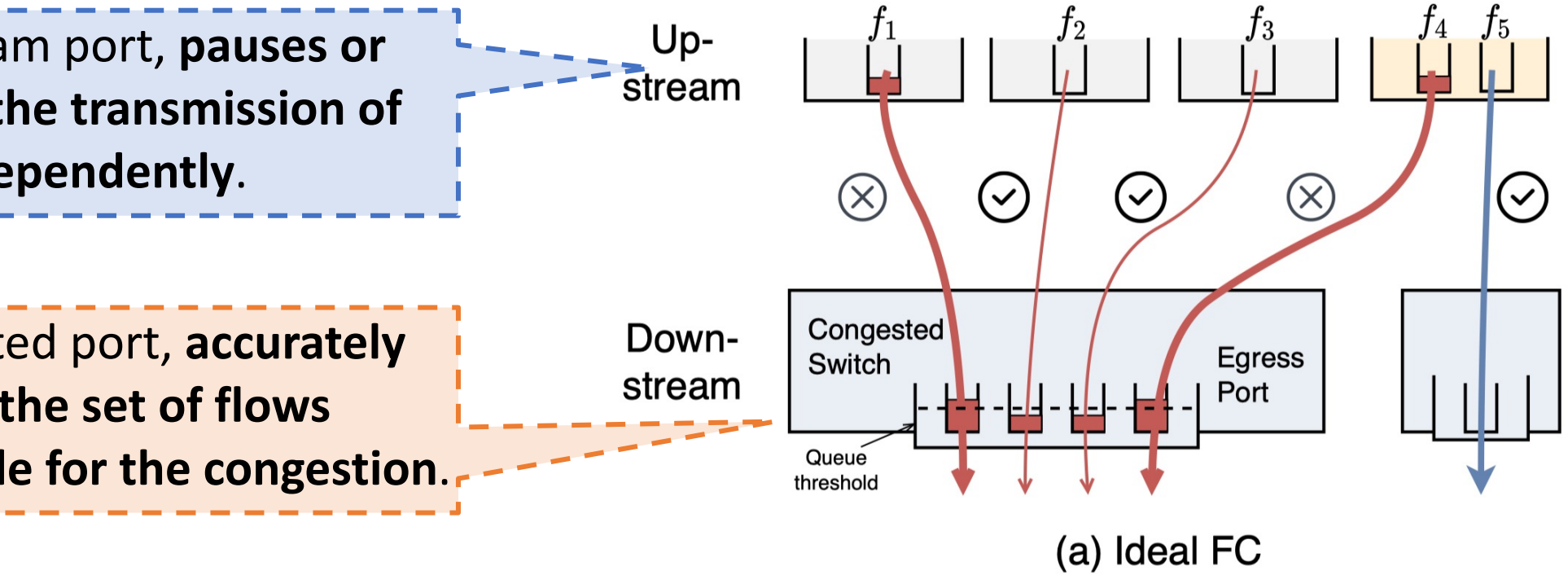
(a) Tail for short flows (b) Avg for large flows

Opportunities of FlowSail

- Efficacy of the ideal FC comes from two key aspects.

At upstream port, **pauses or resumes the transmission of flows independently**.

At congested port, **accurately identifies the set of flows responsible for the congestion**.



(a) Ideal FC

Opportunities of FlowSail

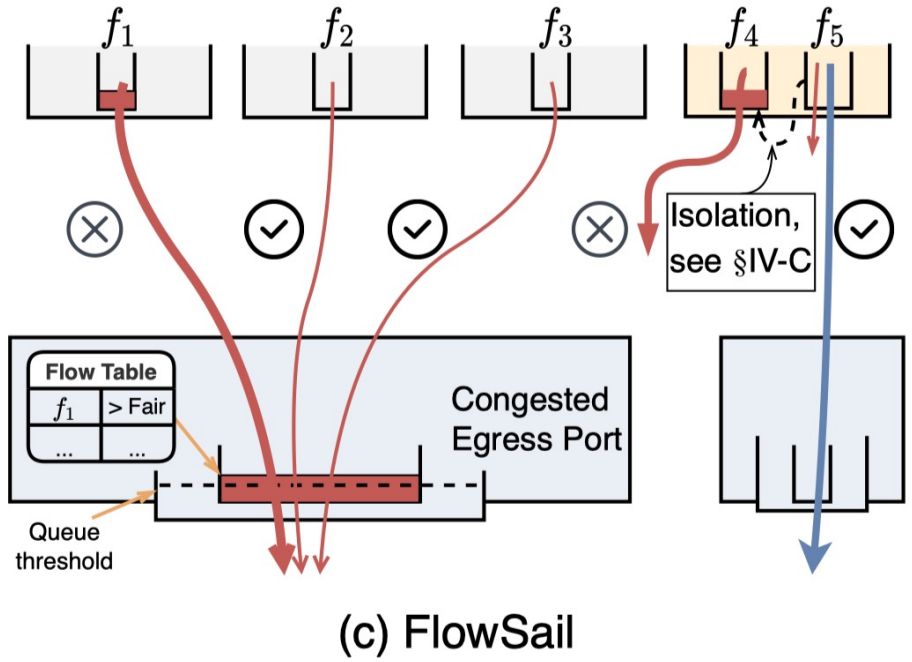
- Opportunities: **it is possible to approximate the behavior of the ideal FC without requiring per-flow queues**.

Redirect all flows that need to be paused to a single paused queue while maintaining other queues active.

Up-stream

Make control decisions at the per-flow level by **monitoring each flow's buffer size**.

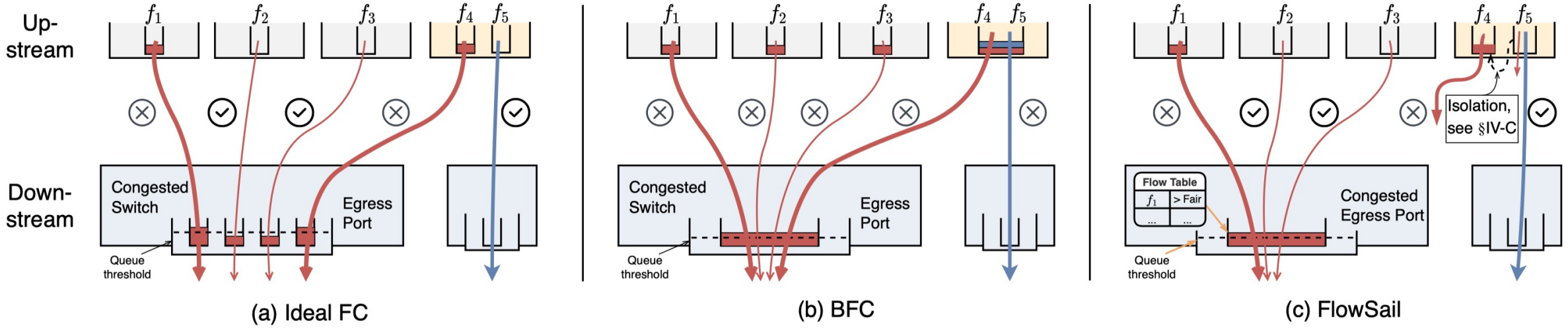
Down-stream



(c) FlowSail

Comparison between BFC and FlowSail

- BFC: manages all flows within the same queue collectively, leading to an unfair degradation of f2, f3, f5.
- FlowSail: approximate the per-flow level granularity (ideal FC).

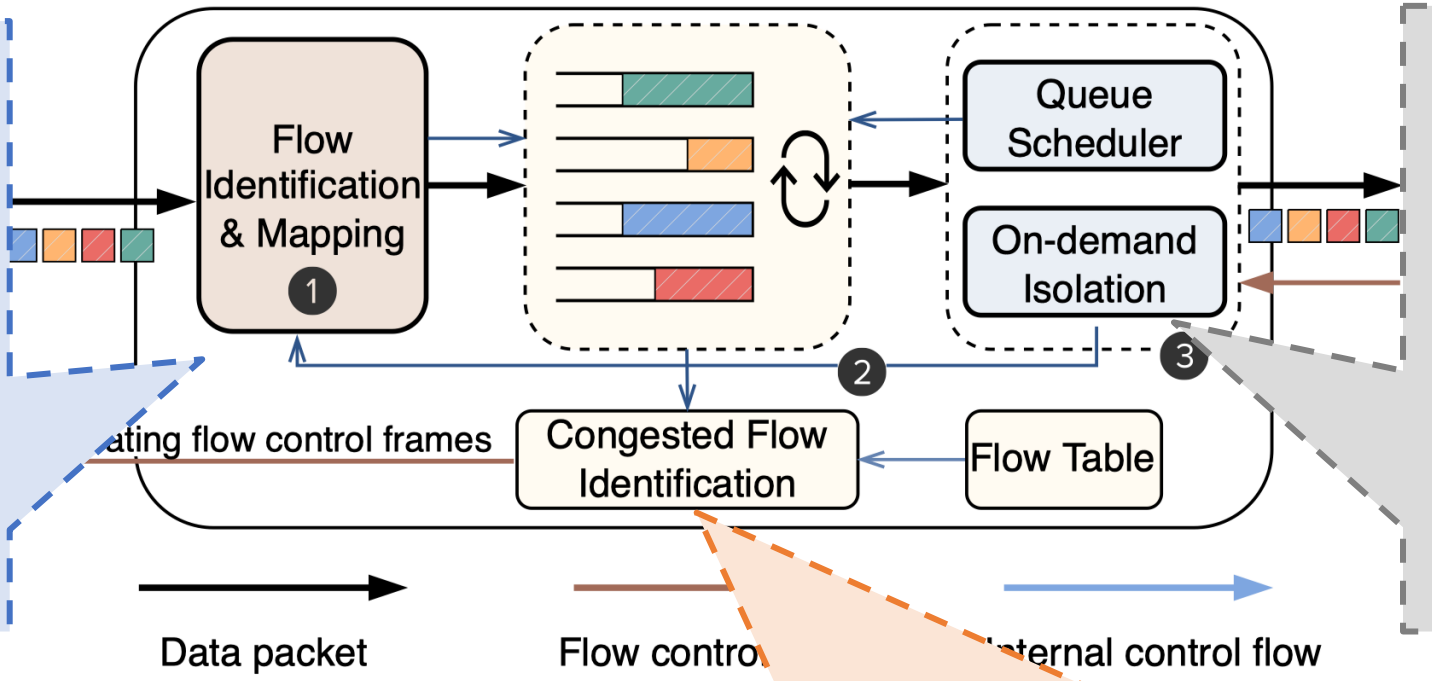


Design of FlowSail

#1. Define flows at the granularity of **active** flows;

Dynamically maps flows to available queues.

(similar with BFC)



#3. **re-assign congested flows** and guarantee in-order delivery.

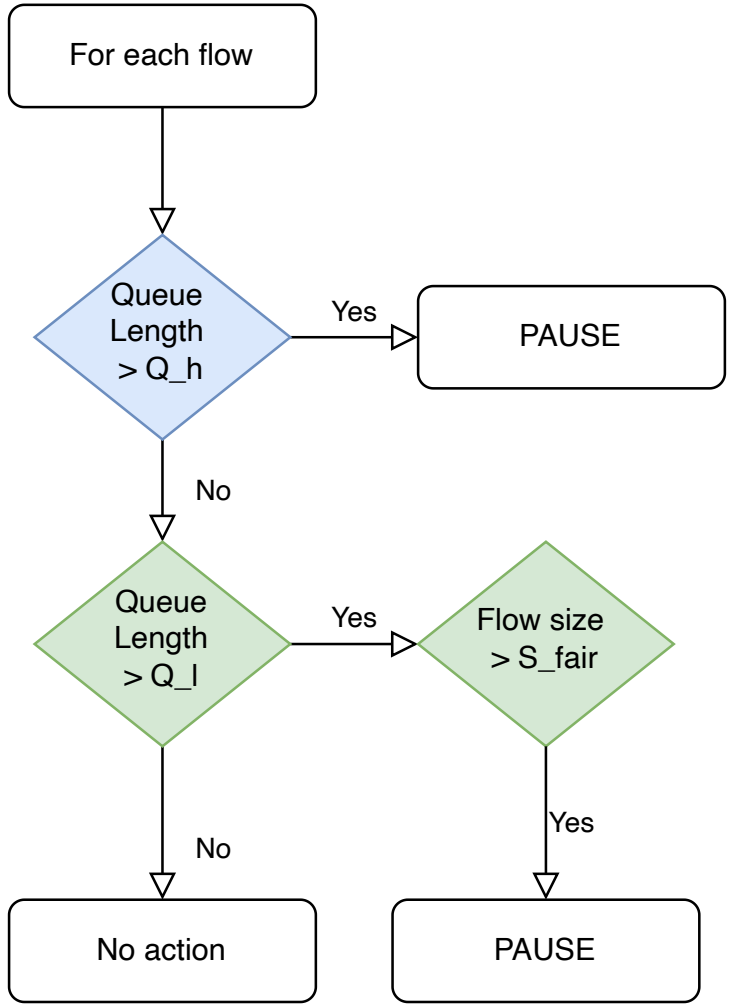
#2. Hierarchical: egress queue length is the primary congestion signal; Flow size is in a secondary level decision.

Hierarchical Congested Flow Identification

- $Q_l < Q < Q_h$: FlowSail only sends PAUSE to flow that occupies more than S_{fair} .
- Hardware-friendly shifting operation and logarithm (counting the number of nonzero bits of data)

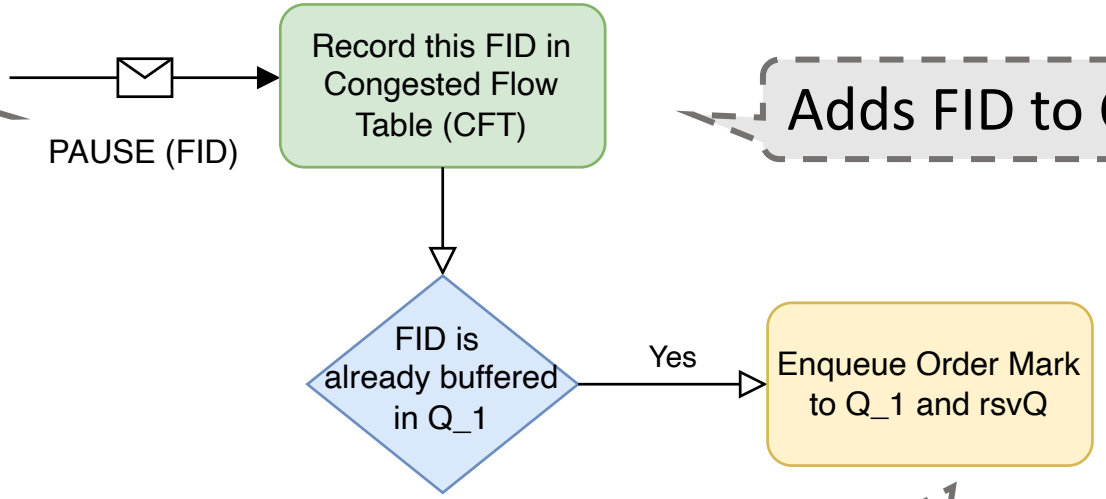
$$S_{fair} = Q >> \lceil \log_2(QT[qIdx].flowNum) \rceil$$

- $Q > Q_h$: FlowSail pauses all passing flows to avoid severe buffer overflow.



On-demand Isolation

PAUSE (RESUME) carries the FID of a congested flow

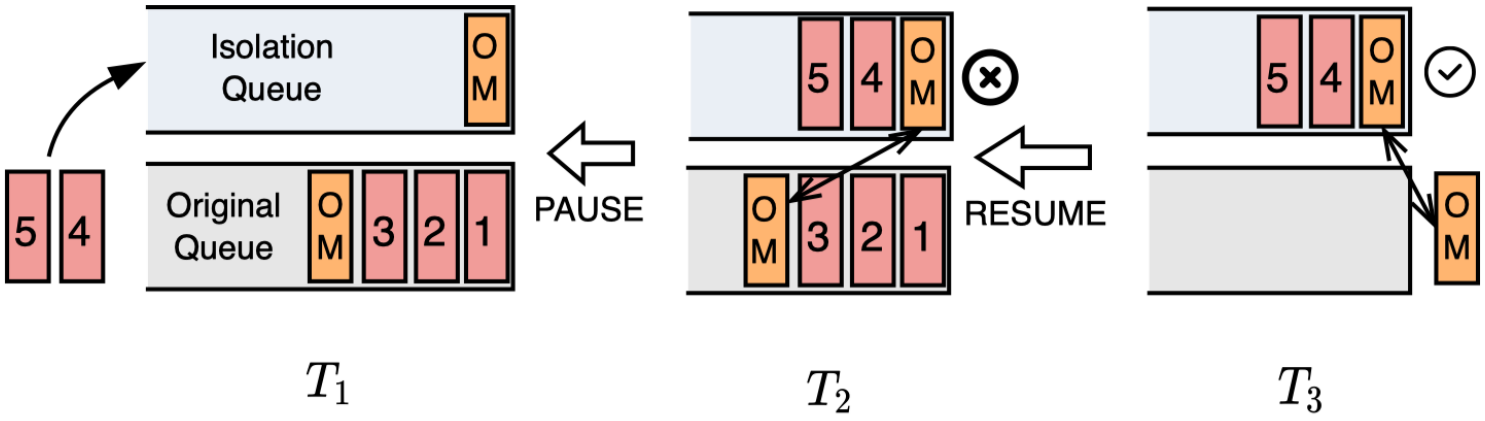


Isolate congested flow to a reserved queue (*rsvQ).*

If flow has buffered packets in original queue (e.g., q1): enqueues an *Order Mark Pair* to q1 and rsvQ and ensure in-order delivery via *Order Mark Matching.*

Order Mark Matching

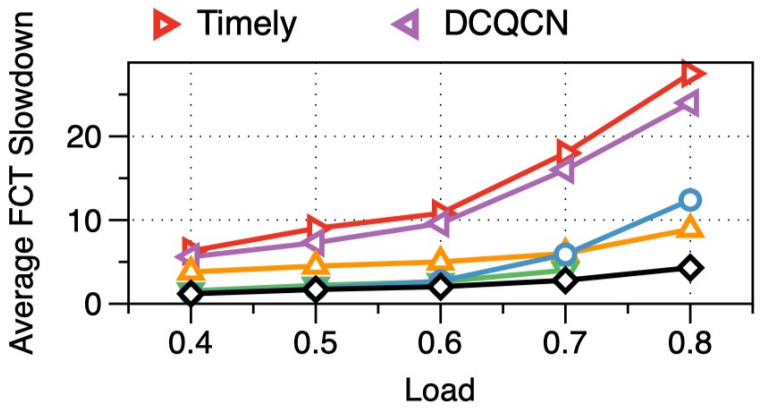
- Every pair of Order Mark (OM) is unique (OM carries FID).
- The OM packet in rsvQ must wait for another matched OM before its transmission begins.



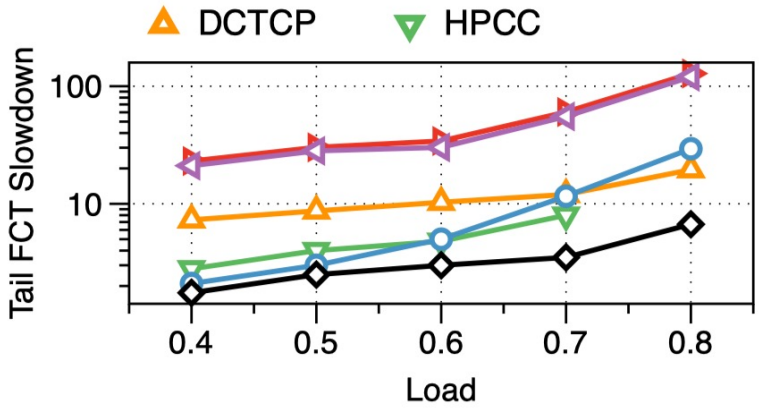
- Implementations details in the paper.

Evaluation: Web Server Distribution

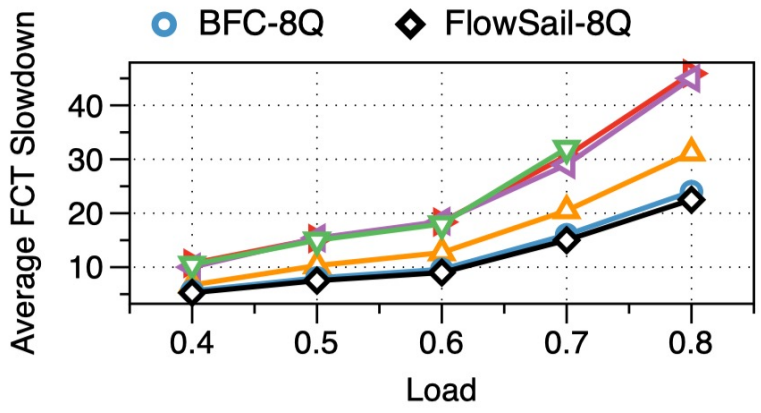
- Short flows (< 10KB) and large flows (> 100KB).
- FlowSail **outperforms BFC (4.3×) and all end-to-end CCs** (e.g., 3.2× compared to DCTCP) in terms of latency for short flows.
- Similar performance in throughput.



(a) Average FCT slowdown for short flow



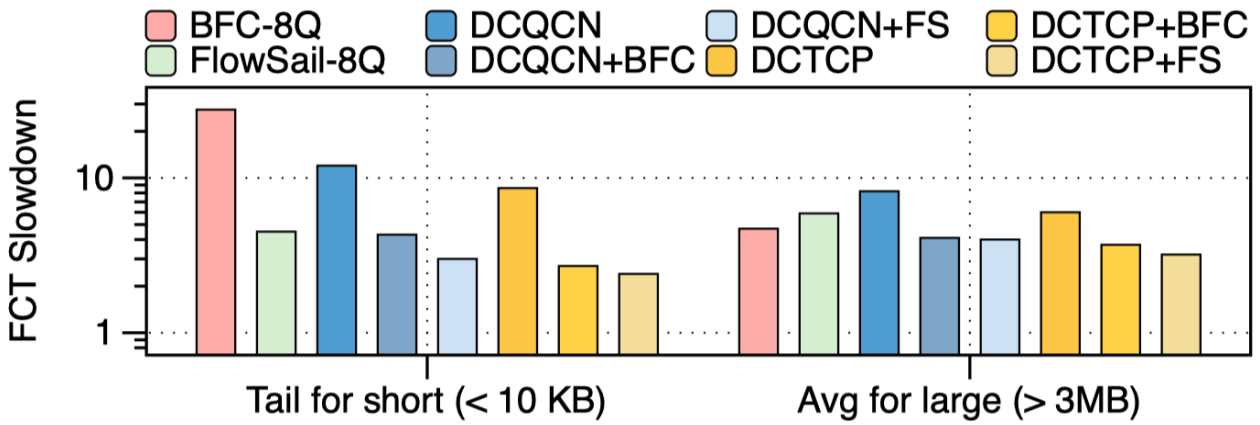
(b) Tail FCT slowdown for short flow

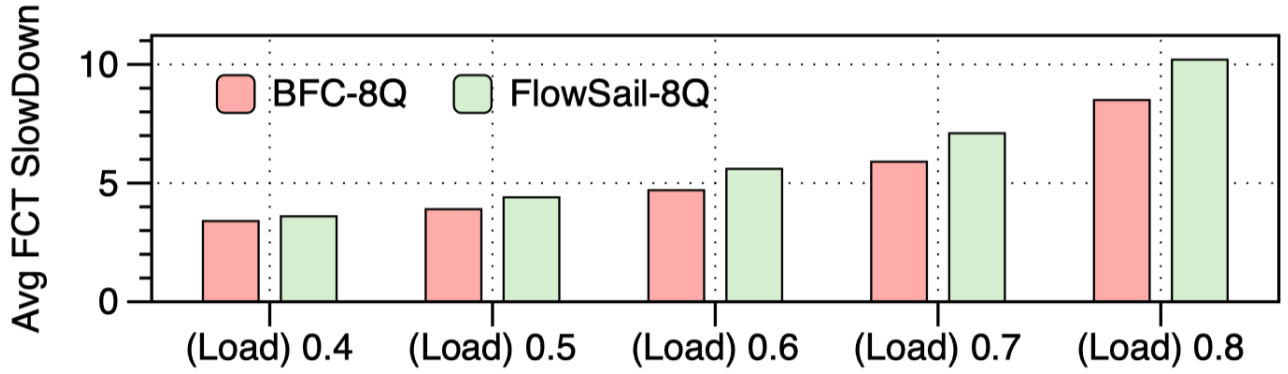


(c) Average FCT slowdown for large flow

Evaluation: Web Search Distribution

- Web Search has more large flows.
- a fixed 60% load: **2.7× reduction in latency of short flow**; throughput decreases for large flows.
- Various loads: no obvious throughput reduction
  ➔ **no absolute trade-off between throughput and latency**.





Conclusion

- FlowSail is a fine-grained flow control scheme at the **per-flow granularity without the requirement of per-flow queues**.
- The core of FlowSail is to effectively **approximate the ideal FC's behavior** at both the congested port and upstream port.
- FlowSail **benefits short flows primarily** without trading off large flows' throughput.

Thank you!

contact: wlicv@connect.ust.hk