FAB: Towards Flow-aware Buffer Sharing in Programmable Switches

Maria Apostolaki

Joint work with
Laurent Vanbever & Manya Ghobadi
An old story...
An old story…
Fan-in causing queue built-up
An old story…
Fan-in causing **drops**
An old story...

Drops increase Flow Completion Times

Drops

Switch

High tail FCT

SLA violations
An old story...
Multiple approaches to address this issue
An old story…
Multiple approaches to address this issue

Active Queue Management

Switch

Drops
An old story...
Multiple approaches to address this issue

- Active Queue Management
- TCP Version

Switch

Drops
An old story...
Multiple approaches to address this issue

- Active Queue Management
- TCP Version
- Scheduling
**Buffer management**: the algorithm according to which ports/queues of a device share a common buffer
Buffer management: the algorithm according to which ports/queues of a device share a common buffer.

Most of today’s devices have a shared buffer to absorb bursts.
How many packets can each port store in the common buffer?
How many packets can each port store in the common buffer?

remaining excessive packets will be dropped
How many packets can each port store in the common buffer?

Let’s give... half of the buffer to each port!

...small fraction to each port!

remaining excessive packets will be dropped.
FAB: Towards Flow-aware Buffer Sharing in Programmable Switches

Joint work with
Laurent Vanbever & Manya Ghobadi
Outline

- Background
- FAB
- Initial Results
- Practicality
Outline

Background

FAB

Initial Results

Practicality
Three common buffer management techniques with pros & cons
Three common buffer management techniques with pros & cons

**Complete Partitioning:**
- statically allocated buffer space per port

---

Complete Sharing:
- unrestricted buffer space per port

---

Dynamic Sharing:
- fraction ($\alpha$) of remaining buffer per port

---

benefits long flows
adapts fairly to load
ignores queue content
works for unbalanced traffic
wastes buffer otherwise
works for balanced traffic
Three common buffer management techniques with pros & cons

**Complete Partitioning:**
statically allocated buffer space per port

**Complete Sharing:**
unrestricted buffer space per port
Three common buffer management techniques with pros & cons

**Complete Partitioning:**
statically allocated buffer space per port

**Complete Sharing:**
unrestricted buffer space per port

**Dynamic Sharing:**
fraction ($\alpha$) of remaining buffer per port
Back to our story…
Shared buffer can host up to 180 packets
Long flows will consume as much buffer as there is available
Short flows will attempt to instantaneously store at most 75 packets in the shared buffer.
Three common buffer management techniques with pros & cons

**Complete Partitioning:**
statically allocated buffer space per port

**Complete Sharing:**
unrestricted buffer space per port

**Dynamic Sharing:**
\[ \text{fraction } (\alpha) \text{ of remaining buffer per port} \]
Complete Partitioning: static buffer space per port = 10 packets
Receiver of long flows buffers up to 10 packets
Receiver of long flows buffers up to 10 packets
Receiver of long flows buffers up to 10 packets

Unused 95%

Port 1 used 5%
Receiver of short flows buffers up to 10 packets
Receiver of short flows buffers up to 10 packets
Buffer is 90% empty

- Unused 90%
- Used by port 2
Buffer is 90% empty, *yet the burst is not absorbed*
Three common buffer management techniques with pros & cons

**Complete Partitioning:**
statically allocated buffer space per port

- ✓ works for balanced traffic
- ✗ wastes buffer otherwise

**Complete Sharing:**
unrestricted buffer space per port

**Dynamic Sharing:**
fraction ($\alpha$) of remaining buffer per port
Three common buffer management techniques with pros & cons

**Complete Partitioning:**
statically allocated buffer space per port

**Complete Sharing:**
unrestricted buffer space per port

**Dynamic Sharing:**
fraction ($\alpha$) of remaining buffer per port
Complete Sharing: unrestricted buffer space per port
Long flows use all buffer
Long flows use all buffer
Long flows use all buffer

![Graph showing the number of packets in the buffer over time. The graph indicates that long flows use all the buffer.](image_url)
Upon arrival, the burst finds the buffer fully occupied
The buffer is fully utilized

# Pkts in Buffer

Time (ns)

0 20 40 60 80 100 120 140 160 180 200

0 2x10^8 4x10^8 6x10^8 8x10^8 1x10^9 1.2x10^9

10% 90%

Pie chart showing 10% and 90% utilization.
The buffer is fully utilized, **but the burst is not absorbed**
Three common buffer management techniques with pros & cons

**Complete Partitioning:**
statically allocated buffer space per port

**Complete Sharing:**
unrestricted buffer space per port

**Dynamic Sharing:**
fraction ($\alpha$) of remaining buffer per port

- ✓ works for unbalanced traffic
- ✗ benefits continuous buffer use
Three common buffer management techniques with pros & cons

**Complete Partitioning:**
statistically allocated buffer space per port

**Complete Sharing:**
unrestricted buffer space per port

**Dynamic Sharing:**
fraction ($\alpha$) of remaining buffer per port
Dynamic Sharing: fraction ($\alpha$) of remaining buffer per port

$$\text{Limit per port} = \frac{\alpha}{\alpha n + 1} N$$

$n$: Number of congested ports
$N$: Buffer Size (packets/Bytes)
$\alpha$: per port/queue parameter
Dynamic Sharing: a fraction ($\alpha$) of remaining shared buffer per port
One port uses buffer.

Limit = \frac{\alpha}{\alpha n + 1} \quad N = \frac{1}{1 + 1} \quad 180 = 90
One port uses buffer.

Limit = \( \frac{\alpha}{\alpha n + 1} \)  

\( N = \frac{1}{1 + 1} \)  

180 = 90
One port uses buffer.

Limit: \( \frac{\alpha}{\alpha n + 1} \quad N = \frac{1}{1 + 1} \quad 180 = 90 \)
Two port uses buffer.

Limit = \( \frac{\alpha}{\alpha n + 1} \)

N = \( \frac{1}{2 + 1} \)

180 = 60
Two port uses buffer.

Limit = \frac{\alpha}{\alpha n + 1} \quad N = \frac{1}{2 + 1} \quad 180 = 60
33% of the buffer is empty,
33% of the buffer is empty, yet the burst is not fully absorbed.
Three common buffer management techniques with pros & cons

Complete Partitioning:
statically allocated buffer space per port

Complete Sharing:
unrestricted buffer space per port

Dynamic Sharing:
fraction (\(\alpha\)) of remaining buffer per port

✓ adapts fairly to load
✗ ignores queue content
Why should we care about queue content?
Why should we care about queue content?

Short flows would benefit more from buffer
Outline

Background

FAB

Initial Results

Practicality
FABULOUS Sharing (FAB) improves dynamic sharing by
Flow-aware Buffer Sharing (FAB) improves dynamic sharing by
Flow-aware Buffer Sharing (FAB) improves dynamic sharing by

- using multiple $\alpha$ per queue/port
  
  Two packets of same ingress and egress port which arrived together, might see different limits
Flow-aware Buffer Sharing (FAB) improves dynamic sharing by

- using multiple $\alpha$ per queue/port
  Two packets of same ingress and egress port which arrived together, might see different limits

- mapping packets to an $\alpha$ according to priority
  $\alpha$ is proportionate to the per-flow expected benefit from buffering
Flow-aware Buffer Sharing (FAB) improves dynamic sharing by:

- using multiple $\alpha$ per queue/port
  Two packets of same ingress and egress port which arrived together, might see different limits

- mapping packets to an $\alpha$ according to their priority in buffering
  $\alpha$ is proportionate to the per-flow expected benefit from buffering

- deciding packet’s priority directly in the data plane
  e.g. prioritizing packets based on flow size
Flow-aware Buffer Sharing (FAB)

\(\alpha = 0.1\) for long flows
\(\alpha = 10\) for short flows
Flow-aware Buffer Sharing (FAB)
FAB maps long flows to $\alpha = 0.1$
Limit for long flows = \( \frac{\alpha}{\alpha n + 1} \)

\[ N = \frac{0.1}{0.1 + 1} \]

180 = 16
Limit for long flows $= \frac{\alpha}{\alpha n + 1}$

$N = \frac{0.1}{0.1 + 1}$

$180 = 16$
Limit for long flows = \( \frac{\alpha}{\alpha n + 1} \)  
\( N = \frac{0.1}{0.1 + 1} \)

180 = 16
FAB maps short flows to $\alpha=10$
Limit for short flows = \( \frac{\alpha}{\alpha n + 1} \)  

\[ N = \frac{10}{10 \times 2 + 1} \]  

180 = 85
Limit for short flows = \( \frac{\alpha}{\alpha n + 1} \) \quad N = \frac{10}{10 \times 2 + 1} \quad 180 = 85
Limit for short flows = \( \frac{\alpha}{\alpha n + 1} \)

\[ N = \frac{10}{10 \times 2 + 1} \]

\[ 180 = 85 \]
Outline

Background

FAB

Initial Results

Practicality
Simulation Results

- Short-Flows Workload
- Mixed Workload
Short-Flows Workload

Senders

Receiver
Dynamic Sharing does not allow burst to be fully buffer, resulting in increased tail FCT.
Dynamic Sharing does not allow burst to be fully buffer, resulting in increased tail FCT
Mixed Workload

- Short Flows
- Long Flows

Senders
FAB limits long flows and allows the burst to use the buffer.
FAB limits long flows and allows the burst to use the buffer
Outline

Background

FAB

Initial Results

Practicality
Is FAB practical?

- **approximates flow size with flow arrival time**
  
  Use bloom filter to store flows that started in discrete time windows

- **enables dropping at the ingress based on FAB**
  
  Dropping decisions at ingress based on buffer occupancy and flow information

- **configures complete sharing**
  
  Disallow traffic manager to drop any packet as long as there is space in the buffer
Outline

Background

FAB

Initial Results

Practicality
FAB: Flow-aware buffer sharing

Splits the buffer space according to the expected benefit from using it for each flow.

Allocates more buffer to short flows, which are distinguished in the data plane.

Significantly decreases flow completion time.