Extending SDN to the Data Plane

Anirudh Sivaraman, Keith Winstein, Suvinay Subramanian,
Hari Balakrishnan

M.I.T.

http://web.mit.edu/anirudh/www/sdn-data-plane.html

November 7, 2013

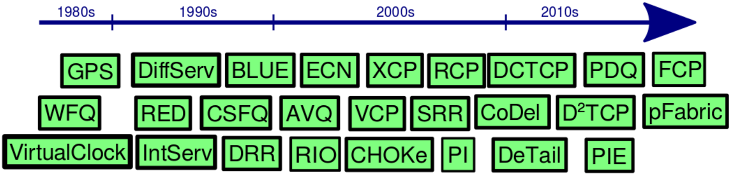
Switch Data Planes today

Two key decisions on a per-packet basis:

- Scheduling: Which packet should be transmitted next?
- Queue Management: How long can queues grow? Which packet to drop?

The Data Plane is continuously evolving

- The long lineage of in-network algorithms:



- Each scheme wins in its own evaluation.

- Some believe in a “silver bullet” knobless in-network method.

We disagree: There is no silver bullet!

- Different applications care about different objectives.
- Applications use different transport protocols.
- Networks are heterogeneous.

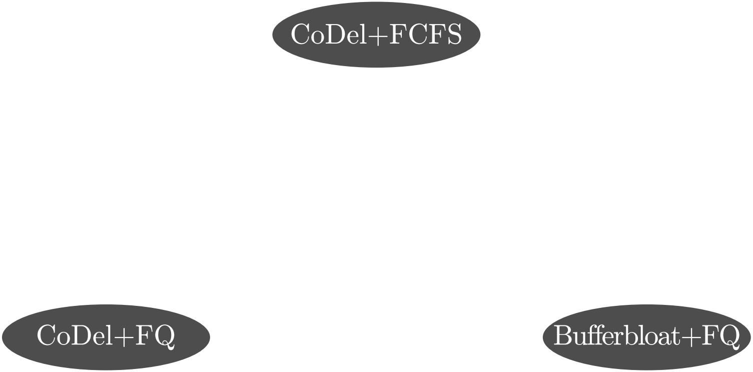
Quantifying “No Silver Bullet”: Network Configurations

Configuration	**Description**
CoDel+FCFS	One shared FCFS queue with CoDel
CoDel+FQ	Per-flow fair queueing with CoDel on each queue
Bufferbloat+FQ	Per-flow fair queueing with deep buffers on each queue

Quantifying “No Silver Bullet”: Workloads and Objectives

Workload	Description	Objective
Bulk	Long-running TCP flow	Maximize throughput
Web	Switched TCP flow with ON and OFF periods	Minimize 99.9 %ile flow completion time
Interactive	Long-running interactive application	Maximize $\frac{\text{throughput}}{\text{delay}}$, i.e., “power”

Quantifying “No Silver Bullet”

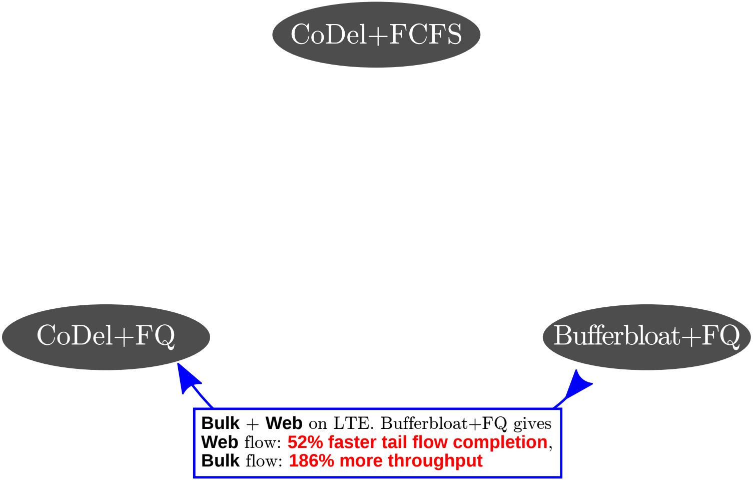


CoDel+FCFS

CoDel+FQ

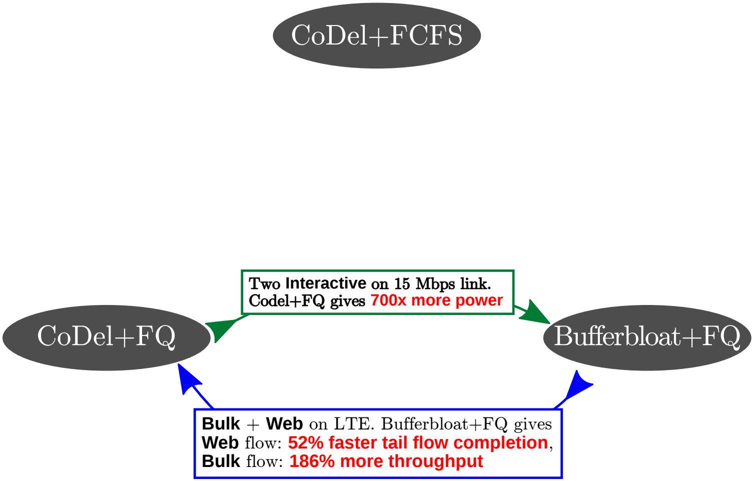
Bufferbloat+FQ

Quantifying “No Silver Bullet”



Bulk + **Web** on LTE. Bufferbloat+FQ gives
Web flow: **52% faster tail flow completion**,
Bulk flow: **186% more throughput**

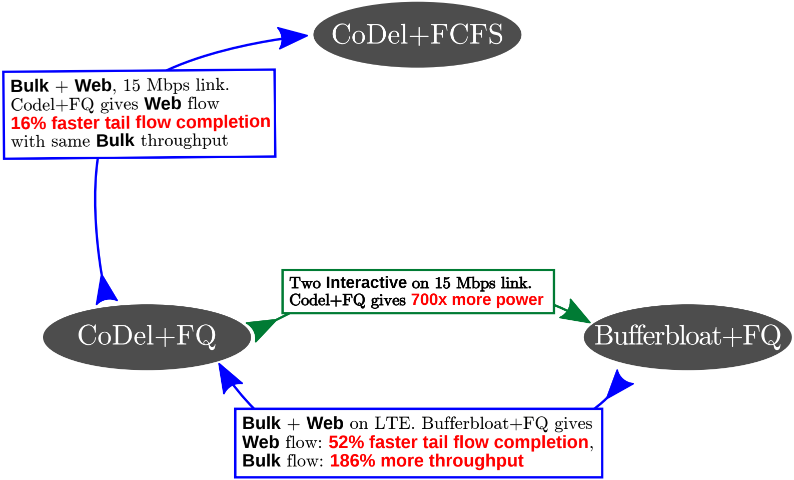
Quantifying “No Silver Bullet”



Two **Interactive** on 15 Mbps link.
Codel+FQ gives **700x more power**

Bulk + **Web** on LTE. Bufferbloat+FQ gives
Web flow: **52% faster tail flow completion**,
Bulk flow: **186% more throughput**

Quantifying "No Silver Bullet"



CoDel+FCFS

Bulk + **Web**, 15 Mbps link.
Codel+FQ gives **Web** flow
16% faster tail flow completion
with same **Bulk** throughput

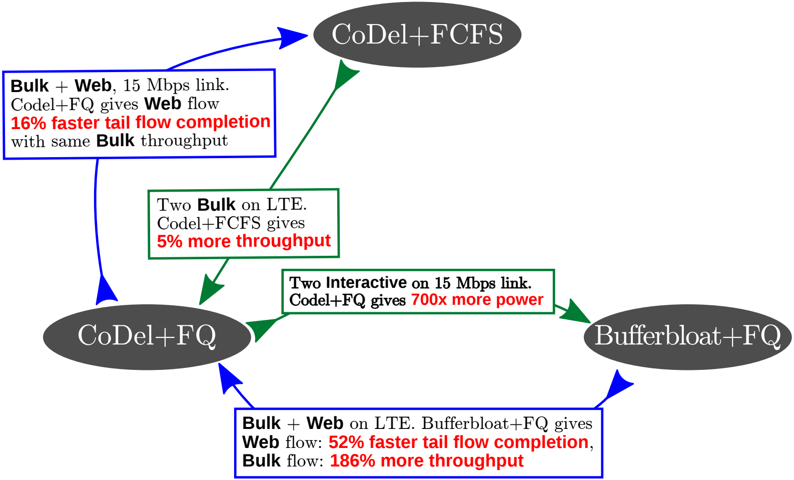
CoDel+FQ

Two Interactive on 15 Mbps link.
Codel+FQ gives 700x more power

Bufferbloat+FQ

Bulk + **Web** on LTE. Bufferbloat+FQ gives
Web flow: **52% faster tail flow completion**,
Bulk flow: **186% more throughput**

Quantifying “No Silver Bullet”



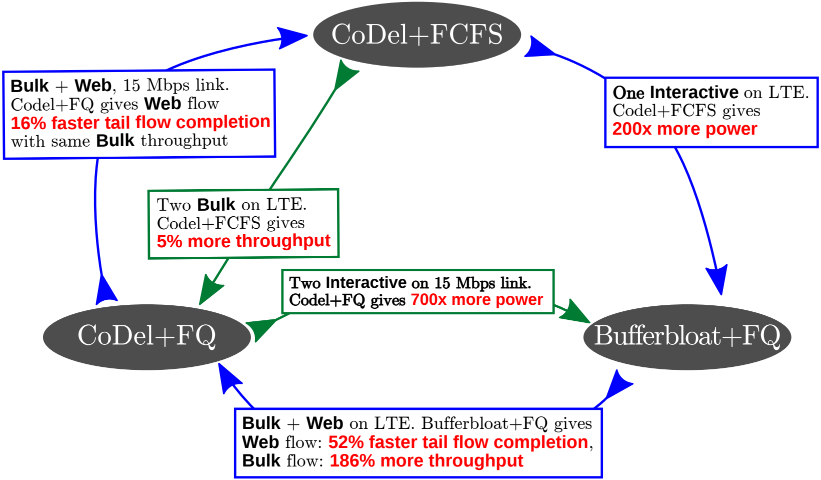
Bulk + **Web**, 15 Mbps link.
Codel+FQ gives **Web** flow
16% faster tail flow completion
with same **Bulk** throughput

Two **Bulk** on LTE.
Codel+FCFS gives
5% more throughput

Two Interactive on 15 Mbps link.
Codel+FQ gives **700x more power**

Bulk + **Web** on LTE. Bufferbloat+FQ gives
Web flow: **52% faster tail flow completion**,
Bulk flow: **186% more throughput**

Quantifying “No Silver Bullet”



CoDel+FCFS

CoDel+FQ

Bufferbloat+FQ

Bulk + **Web**, 15 Mbps link.
Codel+FQ gives **Web** flow
16% faster tail flow completion
with same **Bulk** throughput

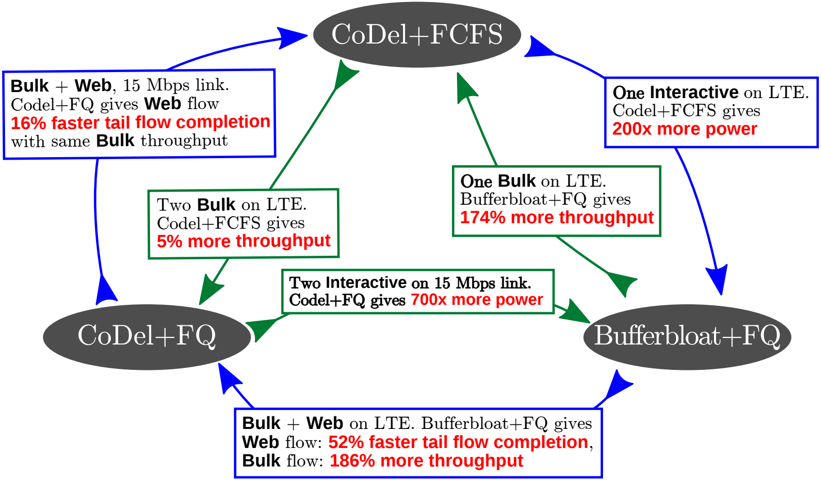
One Interactive on LTE.
Codel+FCFS gives
200x more power

Two **Bulk** on LTE.
Codel+FCFS gives
5% more throughput

Two Interactive on 15 Mbps link.
Codel+FQ gives 700x more power

Bulk + **Web** on LTE. Bufferbloat+FQ gives
Web flow: **52% faster tail flow completion**,
Bulk flow: **186% more throughput**

Quantifying “No Silver Bullet”



CoDel+FCFS

Bulk + **Web**, 15 Mbps link. Codel+FQ gives **Web** flow **16% faster tail flow completion** with same **Bulk** throughput

One Interactive on LTE. Codel+FCFS gives **200x more power**

Two **Bulk** on LTE. Codel+FCFS gives **5% more throughput**

One Bulk on LTE. Bufferbloat+FQ gives **174% more throughput**

Two Interactive on 15 Mbps link. Codel+FQ gives 700x more power

CoDel+FQ

Bufferbloat+FQ

Bulk + **Web** on LTE. Bufferbloat+FQ gives **Web** flow: **52% faster tail flow completion**, **Bulk** flow: **186% more throughput**

Why is no single data plane configuration the best?

- Bufferbloat on variable-rate links helps throughput!
  - Variable-rate links have an inherent delay-throughput tradeoff
- FCFS is preferable to Fair Queuing in some cases
  - When equally aggressive flows compete, they don't need protection from each other
  - Helps reduce tail packet delay
- Fair Queuing is required in some cases
  - When competing flows aren't equally aggressive, isolation helps

So what should the network designer do?

Architect a flexible data plane

- Programmable queue management and scheduling
- Not just for selecting among pre-built choices, but to change behavior in the field
- Because there is no silver bullet and innovation will continue!

Controlled flexibility: Want performance, security

(Or, why this isn't the same as "active networks")

- Provide interfaces only to the head and tail of queues
- Operators specify only queue-management/scheduling logic
- No access to packet payloads (for now)

Building such a data plane in four parts

- Hardware gadgets
  - Random number generators (RED, BLUE)
  - Binary tree of comparators (pFabric, SRPT)
- I/O interfaces
  - Drop/mark head/tail of queue
  - Interrupts for enqueue/dequeue
- State maintenance
  - Per-flow (WFQ, DRR)
  - Per-dst address (PF)
- A domain-specific instruction set
  - Expresses control flow
  - Implements new functions unavailable in hardware

Feasibility study: CoDel

Synthesis numbers on Xilinx Kintex-7:

Resource	**Usage**	**Fraction of FPGA**
Slice logic	1,256	1%
Slice logic dist.	1,975	2%
IO/GTX ports	27	2%
DSP slices	0	0%
Maximum speed	12.9 $\times 10^6$ pkts/s ~10gbps	

- Small fraction of the FPGA's resources.
- Can be improved by pipelining or parallelizing.

Conclusion

- There is no silver bullet to in-network resource control because of application and network diversity
- Algorithms will continue to evolve: the data plane should help
- Directions to reproduce results: http://web.mit.edu/anirudh/www/sdn-data-plane.html

Limitations and Practical Considerations:

- Cannot express several network functions that need payloads.
- How do applications signal objectives to the network?
- Feasibility at 10G on high port-density switches.
- Mechanism to map flows onto per-port queues.
- Energy and Area overheads.