Extending SDN to the Data Plane

Anirudh Sivaraman, Keith Winstein, Suvinay Subramanian, Hari Balakrishnan

M.I.T.

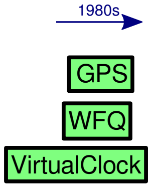
http://web.mit.edu/anirudh/www/sdn-data-plane.html

Switch Data Planes today

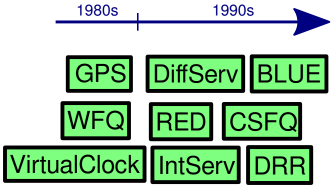
Two key decisions on a per-packet basis:

- **Scheduling**: Which packet to transmit next?
- **Queue Management**: How long can queues grow? Which packet to drop?

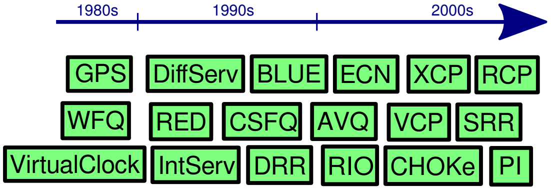
The long lineage of in-network algorithms



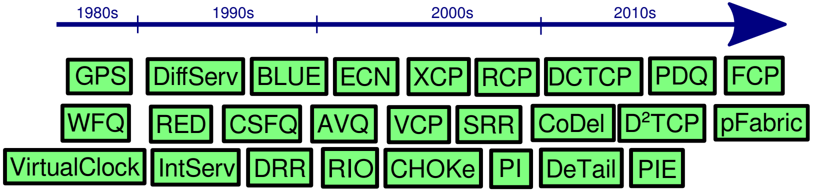
The long lineage of in-network algorithms



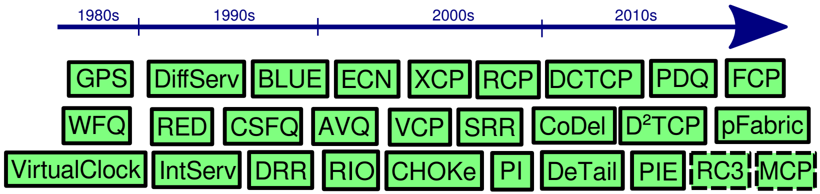
The long lineage of in-network algorithms



The long lineage of in-network algorithms



The long lineage of in-network algorithms



The Data Plane is continuously evolving

- Each scheme wins in its own evaluation.
- Quest for a “silver bullet” in-network method.

We disagree: There is no silver bullet!

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

Our work:

- Quantify non-universality of in-network methods.
- Extend SDN to the Data Plane to handle in-network diversity.

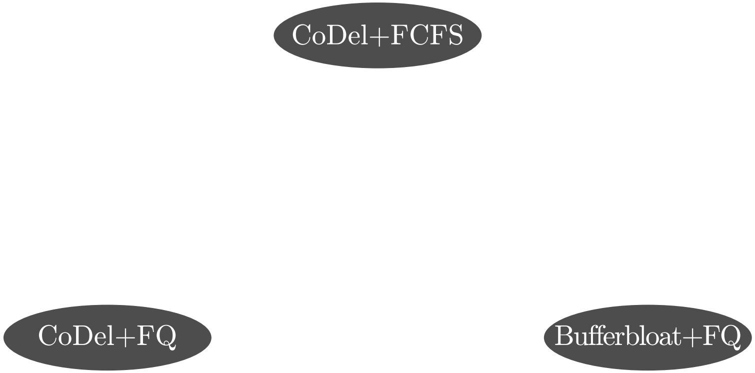
Quantifying "No Silver Bullet": Network Configurations

Configuration	Description
CoDel+FCFS	One shared FCFS queue with CoDel
CoDel+FQ	Per-flow fair queueing with CoDel on each queue (Nichols 2013)
Bufferbloat+FQ	Per-flow fair queueing with deep buffers on each queue

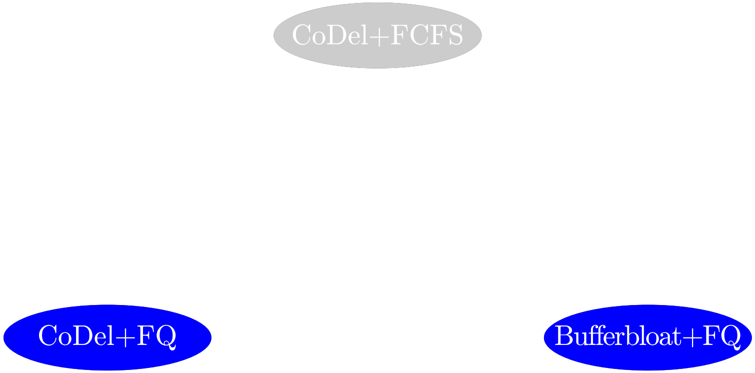
Quantifying "No Silver Bullet": Workloads and Objectives

Workload	**Description**	**Objective**
Bulk	Long-running bulk transfer flow	Max. throughput
Web	Switched flow with ON/OFF periods	Min. 99.9 %ile flow completion time
Interactive	Long-running interactive flow	Max. $\frac{\text{throughput}}{\text{delay}}$

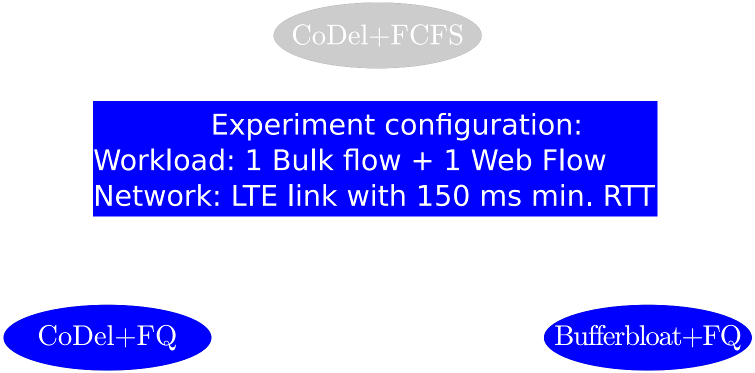
Quantifying “No Silver Bullet”



Quantifying "No Silver Bullet"

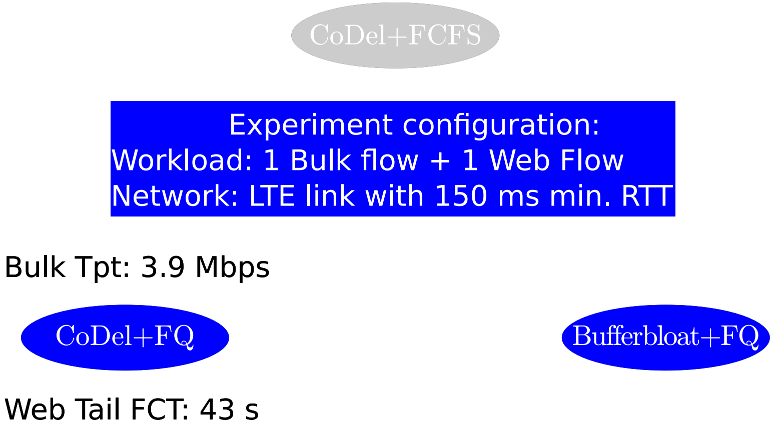


Quantifying “No Silver Bullet”



Experiment configuration:
Workload: 1 Bulk flow + 1 Web Flow
Network: LTE link with 150 ms min. RTT

Quantifying "No Silver Bullet"



Experiment configuration:
Workload: 1 Bulk flow + 1 Web Flow
Network: LTE link with 150 ms min. RTT

Bulk Tpt: 3.9 Mbps

Web Tail FCT: 43 s

Quantifying "No Silver Bullet"

CoDel+FCFS

Experiment configuration:
Workload: 1 Bulk flow + 1 Web Flow
Network: LTE link with 150 ms min. RTT

CoDel+FQ	Bufferbloat+FQ
Bulk Tpt: 3.9 Mbps	Bulk Tpt: 11.2 Mbps
Web Tail FCT: 43 s	Web Tail FCT: 21 s

Quantifying "No Silver Bullet"

CoDel+FCFS

Experiment configuration:
Workload: 1 Bulk flow + 1 Web Flow
Network: LTE link with 150 ms min. RTT

Bulk Tpt: 3.9 Mbps

CoDel+FQ

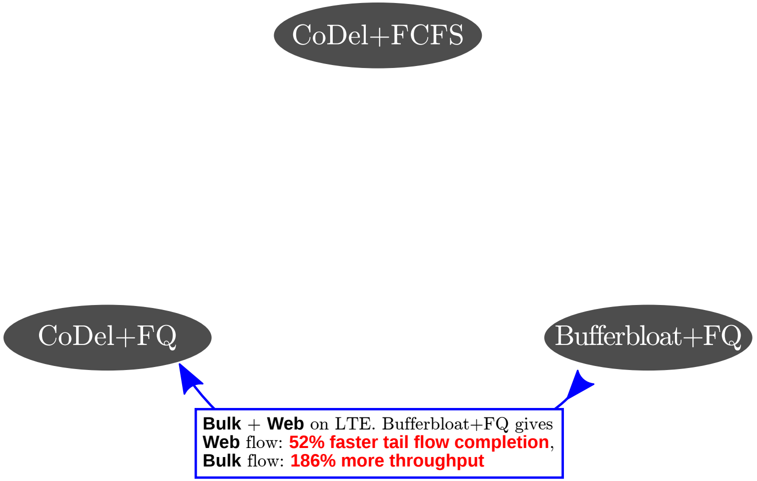
Web Tail FCT: 43 s

Bulk Tpt: 11.2 Mbps

Bufferbloat+FQ

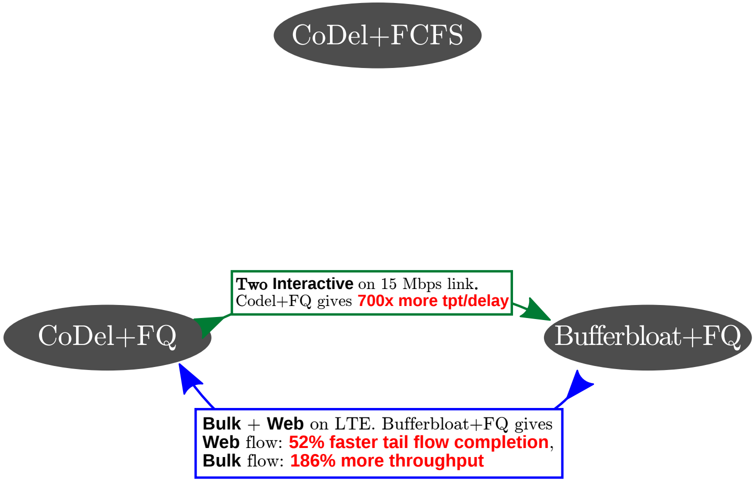
Web Tail FCT: 21 s

Quantifying “No Silver Bullet”



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

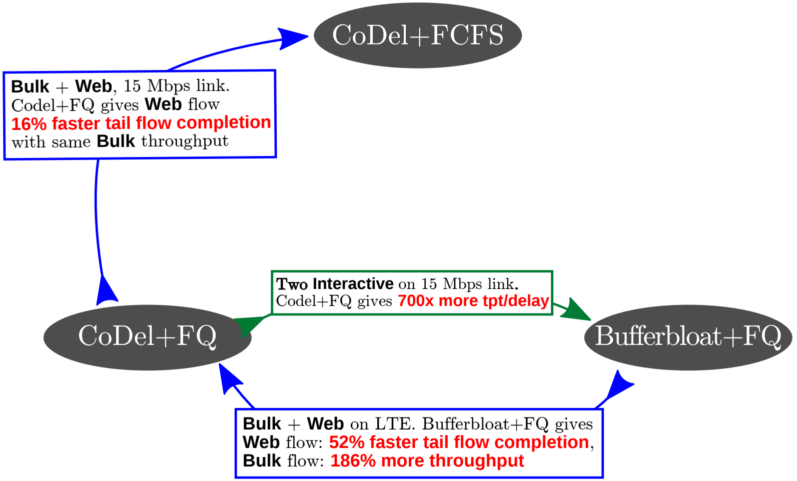
Quantifying "No Silver Bullet"



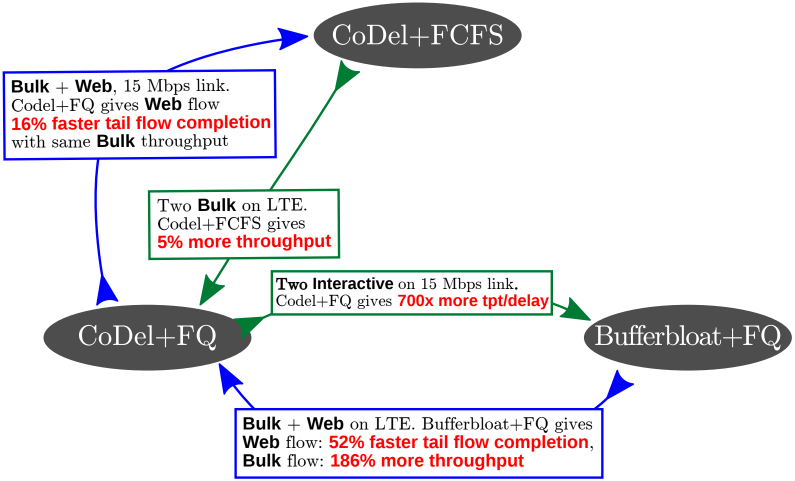
Two Interactive on 15 Mbps link.
Codel+FQ gives **700x more tpt/delay**

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

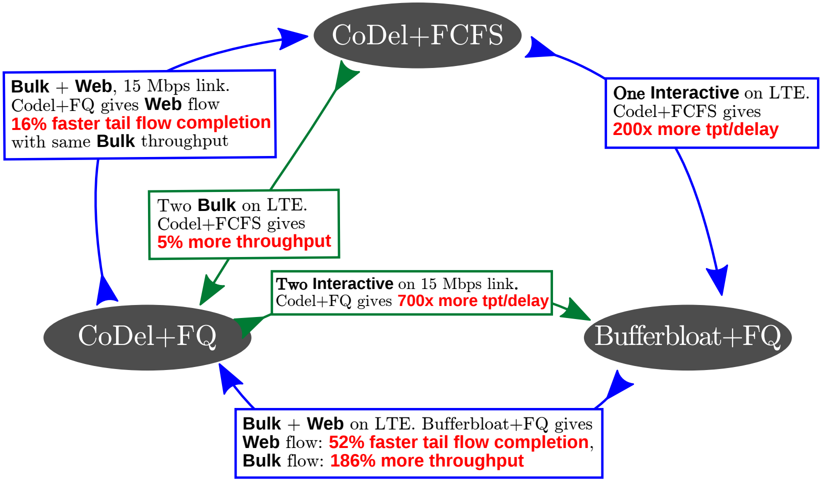
Quantifying “No Silver Bullet”


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

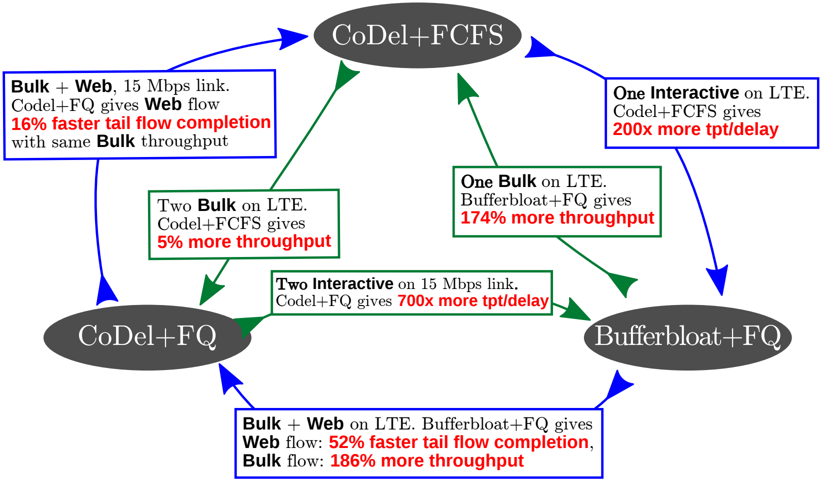
Quantifying "No Silver Bullet"


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
Two **Bulk** on LTE.
Codel+FCFS gives
5% more throughput
Two Interactive on 15 Mbps link.
Codel+FQ gives **700x more tpt/delay**
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 tpt/delay
Two **Bulk** on LTE.
Codel+FCFS gives
5% more throughput
Two Interactive on 15 Mbps link.
Codel+FQ gives **700x more tpt/delay**
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 tpt/delay**
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 tpt/delay**
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 gives the best throughput on variable-rate links.
- FCFS is preferable to Fair Queuing with homogenous objectives.
- Fair Queuing is preferable with heterogeneous objectives.

So what should the network designer do?

- Don't strive for the best in-network behaviour.
- Instead, architect for evolvability.
- Conceptually, extend SDN to include the data plane as well.

Flexibility without sacrificing performance

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

Building such a data plane in four parts

- Hardware gadgets
  - Random number generators (RED, BLUE)
  - Binary tree of comparators (pFabric, SRPT)
  - Look-up tables for function approximation (CoDel, RED)

Building such a data plane in four parts

- Hardware gadgets
  - Random number generators (RED, BLUE)
  - Binary tree of comparators (pFabric, SRPT)
  - Look-up tables for function approximation (CoDel, RED)
- I/O interfaces
  - Drop/mark head/tail of queue
  - Interrupts for enqueue/dequeue
  - Rewrite packet fields

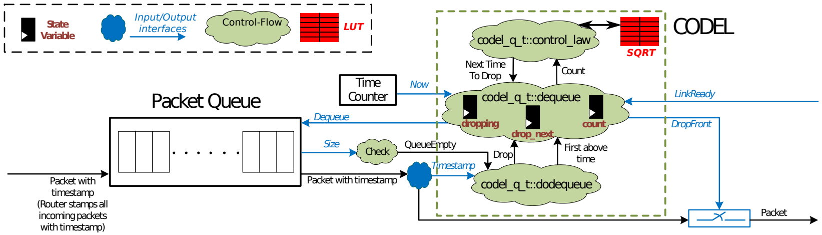
Building such a data plane in four parts

- Hardware gadgets
  - Random number generators (RED, BLUE)
  - Binary tree of comparators (pFabric, SRPT)
  - Look-up tables for function approximation (CoDel, RED)
- I/O interfaces
  - Drop/mark head/tail of queue
  - Interrupts for enqueue/dequeue
  - Rewrite packet fields
- State maintenance
  - Per-flow (WFQ, DRR)
  - Per-dst address (PF)

Building such a data plane in four parts

- Hardware gadgets
  - Random number generators (RED, BLUE)
  - Binary tree of comparators (pFabric, SRPT)
  - Look-up tables for function approximation (CoDel, RED)
- I/O interfaces
  - Drop/mark head/tail of queue
  - Interrupts for enqueue/dequeue
  - Rewrite packet fields
- 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 the Xilinx Kintex-7

Resource	**Usage**	**Fraction**
Slice logic	1,256	1%
Slice logic dist.	1,975	2%
IO/GTX ports	27	2%
DSP slices	0	0%
Maximum speed	12.9 million pkts/s ~10 Gbps	

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

Limitations and Practical Considerations:

- Cannot express several network functions that need payloads.
- Mechanism to signal application objectives.
- Feasibility at 10G on high port-density switches.
- Energy, area, and performance costs of flexibility.

Conclusion

- No silver bullet to in-network resource allocation.
- Algorithms will evolve: Data Plane should help
- Reproduce our results: http://web.mit.edu/anirudh/www/sdn-data-plane.html