U.S. DEPARTMENT OF ENERGY

Elephants Sharing the Highway: Studying TCP Fairness in Large Transfers over High Throughput Links

Imtiaz Mahmud[1], George Papadimitriou[2], Cong Wang[3], Mariam Kiran[4], Anirban Mandal[3], Ewa Deelman[2]

[1]Lawrence Berkeley National Laboratory, [2]University of Southern California, [3]Rennaissance Computing Institute, [4]Oak Ridge National Laboratory

Meet the Team

Ewa Deelman
USC (Lead PI)

Anirban Mandal
RENCI (Co-PI)

Prasanna Balaprakash
ORNL(Co-PI)

Mariam Kiran
ORNL(Co-PI)

George Papadimitriou
USC

Cong Wang
RENCI

Krishnan Raghavan
ANL

Imtiaz Mahmud
LBNL

Komal Thareja
RENCI

Hongwei Jin
ANL

Motivation (1/2)

Scientific data

- Large amount of data is produced in every second
- Needs to be send to cloud - storage / further processing
- Sharing data in the scientific community
- Reliable and timely data delivery is crucial
- Network resources are limited, however, the volume of data is large
- Efficient handling of network resources is the key for successful data delivery

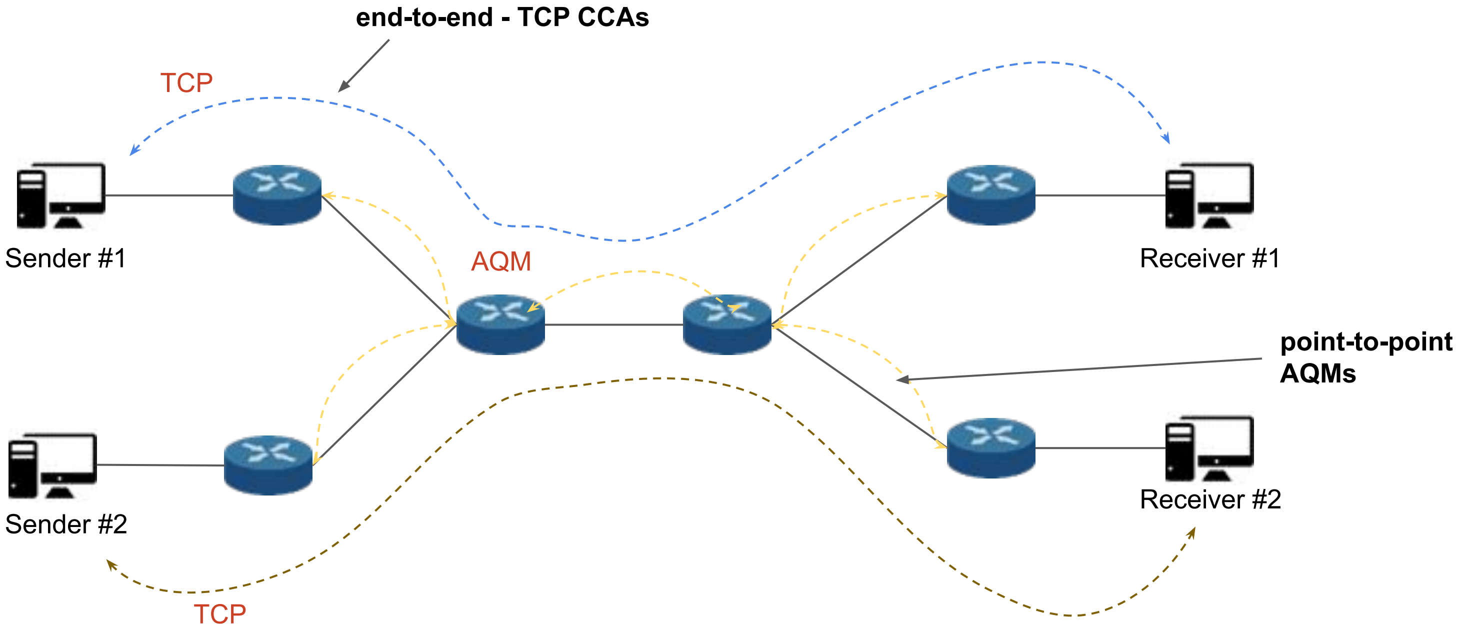
Transmission Control Protocol (TCP)

- Sender to receiver - ensures reliable delivery, fair utilization of the underlying resources, minimize delay
- Wide variant of Congestion Control Algorithms (CCAs)
- Difficult to decide: what, when, and where!!!

Active Queue Management (AQM) algorithms

- Router to router - ensures better network utilization, minimize packet drops, fairness/QoS among flows
- Wide variant of AQM algorithms
- Difficult to decide: what, when, and where!!!

Motivation (2/2)



What are we doing? Improving Network Performance for our data transfers

Reliable data delivery?

TCP Algorithms CCA?

AQM (Queuing)a lgorithm?

Experiment

Novel Innovations for Community:

- Interplay between Transfer and queuing in routers
- Build dataset for transfer protocols behavior
- Lead to a NEW transfer methods for ***"better and fair Internet"***

Background: TCP - BBR, Hamilton, CUBIC, Reno (1/2)

BBRv1

- BBRv1 (Bottleneck Bandwidth and Round-trip propagation time) - developed by Google.
- It aims to maximize network throughput by estimating the available bandwidth and delay of the network path.
- BBRv1 uses a model-based approach to adaptively adjust the sending rate based on the estimated bottleneck bandwidth and round-trip time.

BBRv2

- BBRv2 - improved version of the BBRv1.
- It incorporates several improvements to better handle network congestion and improve fairness.
- BBRv2 includes features like pacing, more accurate congestion signaling and response, and improved performance over high-latency links.

Hamilton TCP (H-TCP)

- H-TCP - designed for data center networks (i.e., high bandwidth low latency networks).
- It focuses on achieving high throughput and low latency in large-scale data center environments.
- Utilizes a combination of explicit congestion control and ECN (Explicit Congestion Notification) feedback to adaptively adjust the sending rate.

Background: BBR, Hamilton, CUBIC, Reno (2/2)

CUBIC

- CUBIC (Compound TCP) is a widely used TCP congestion control algorithm.
- It is designed to provide a fair and efficient sharing of network bandwidth.
- CUBIC utilizes a cubic function to control the TCP sending rate based on the observed network congestion.

Reno

- Reno is one of the oldest and most widely deployed TCP congestion control algorithms.
- It uses a combination of packet loss and TCP timeouts to detect network congestion.
- Reno reduces the sending rate upon congestion signals and gradually increases it when the network is perceived as congestion free.

Background: Active Queue Management (AQM) algorithms

FIFO

- FIFO (First-In-First_Out) is a simple and commonly used queueing algorithm.
- It treats the network queue as a basic buffer and forwards packets in the order they arrived.
- As a point-to-point flow control algorithm, FIFO does not provide any congestion control mechanisms and may lead to buffer bloat and increased latency under heavy network congestion.

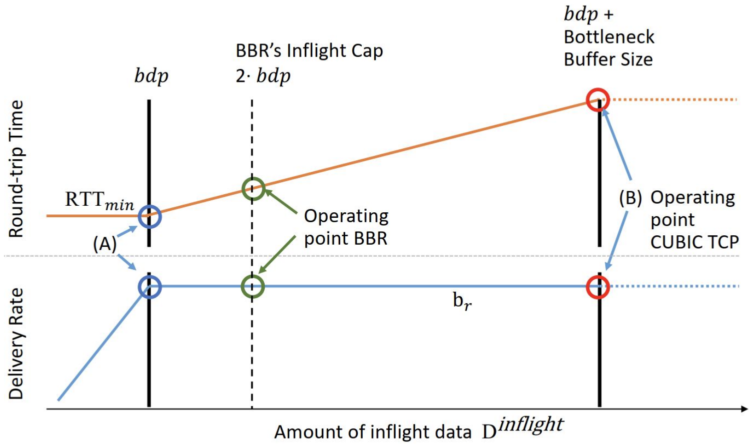
RED

- RED (Random Early Detection) - designed to prevent congestion collapse and improve fairness.
- To control point-to-point network congestion, It randomly drops packets from the queue before the queue becomes completely full, based on configurable thresholds.
- This random drop of pacet signals the end-to-end congestion control algorithms to reduce the flow of data, thus helps maintaining a stable network performance.

FQ_CODEL

- FQ_CODEL (Fair Queueing Controlled Delay) - combines fair queueing and CoDel (Controlled Delay) techniques.
- It assigns separate queues to different flows and manages the queue lengths based on per-flow pacing.
- Fq_codel aims to obtain low latency, low packet loss, and fair bandwidth allocation among flows, particularly in networks with diverse traffic patterns.

Background: BBR vs Cubic with Different Buffer Sizes

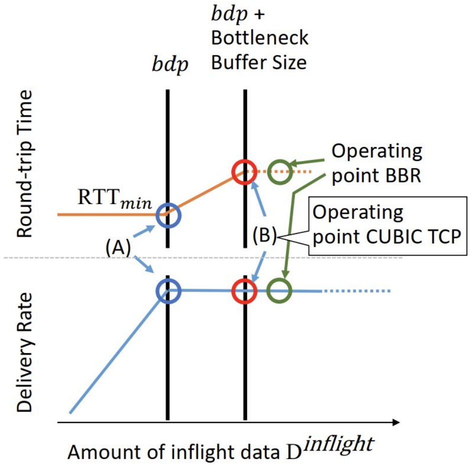


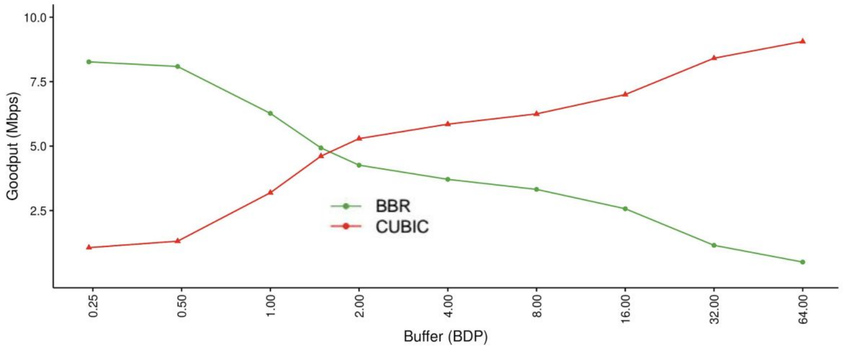
Observed operating points of BBR and Cubic

Ref. Experimental evaluation of BBR congestion control, Mario Hock et al.; ICNP'17, Oct, 2017.

Large buffer

Small buffer



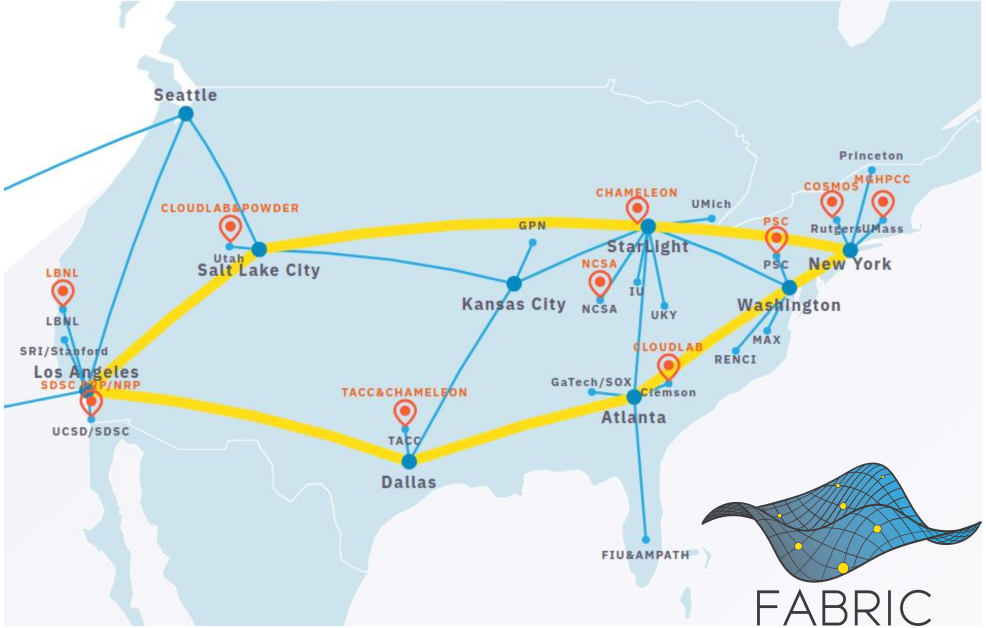


1xCUBIC v 1xBBR goodput: bw=10Mbps, RTT=40ms, 4min transfer, varying buffer sizer

Ref: Google: https://datatracker.ietf.org/meeting/101/materials/slides-101-iccrg-an-update-on-bbr-work-at-google-00

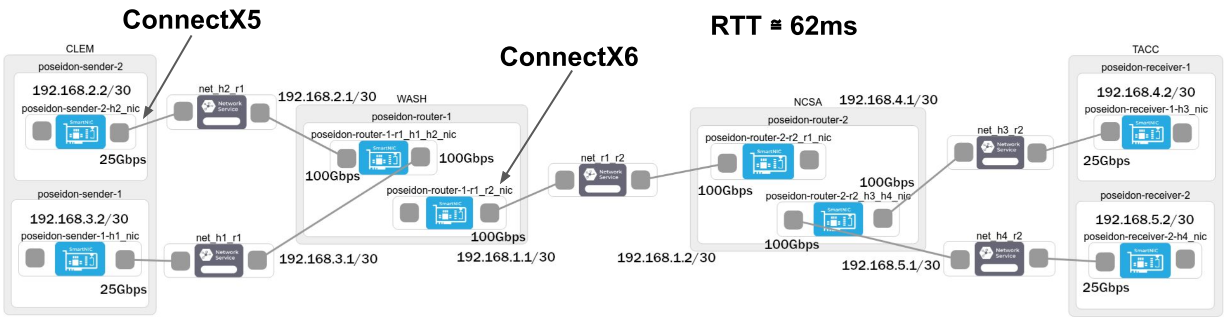
Experimental Setup on FABRIC

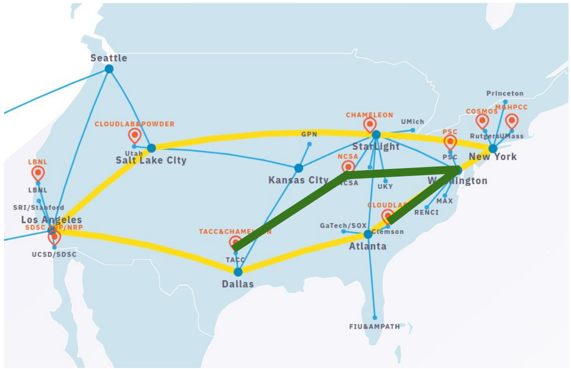
- FABRIC is a nationwide instrument funded by the National Science Foundation (NSF) to enable large scale experimentation.
- FABRIC offers everywhere programmability and provides compute and storage resource in multiple locations, interconnected by high-speed dedicated optical links.
- FABRIC provides a Python API that can be used to design topologies and control the experiments.



https://fabric-testbed.net/

Experimental Setup on FABRIC





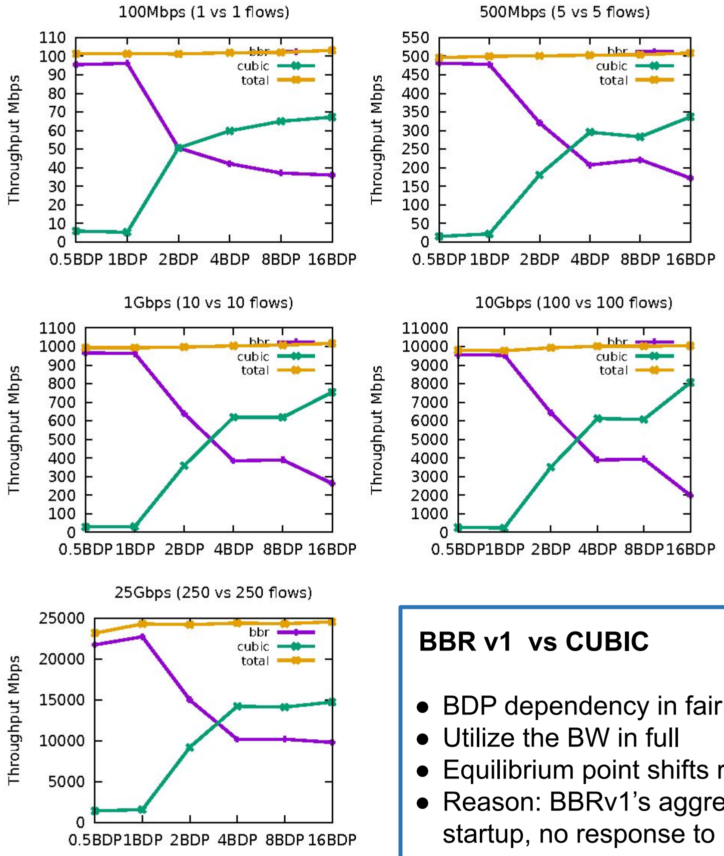
Scenarios

CCA 1 - CCA 2	AQM	Queue Length	Bottleneck BW
BBRv1 - CUBIC			
BBRv2 - CUBIC	FIFO	0.5 x BDP	100 Mbps
HTCP - CUBIC		1 x BDP	500 Mbps
Reno - CUBIC	FQ CODEL	2 x BDP	1 Gbps
CUBIC - CUBIC		4 x BDP	10 Gbps
BBRv1 - BBRv1	RED	8 x BDP	25 Gbps
BBRv2 - BBRv2		16 x BDP	
HTCP - HTCP			
Reno - Reno			

Iperf3 Configuration

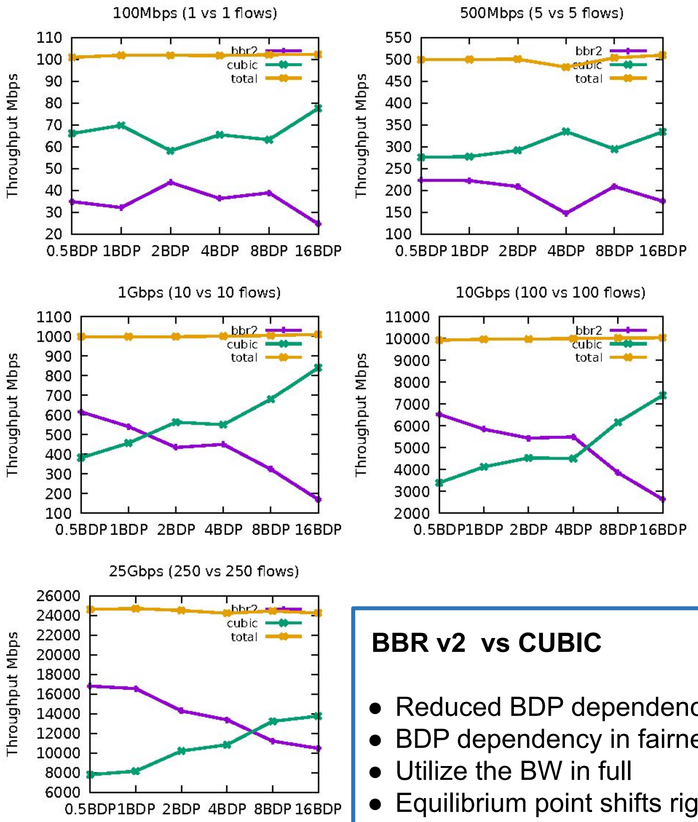
Bottleneck BW	Total #Flows	iperf3 Configuration
100 Mbps	2	1 iperf3 process/node 1 stream
500 Mbps	10	5 iperf3 processes/node 1 stream each
1 Gbps	20	10 iperf3 processes/node 1 stream each
10 Gbps	200	10 iperf3 processes/node 10 parallel streams each
25 Gbps	500	25 iperf3 processes/node 10 parallel streams each

Observation on CCAs when AQM is FIFO (1/2)



BBR v1 vs CUBIC

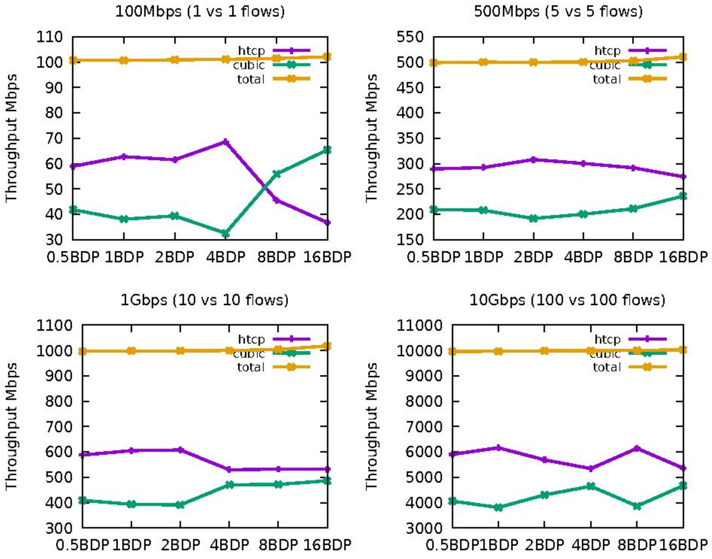
- BDP dependency in fairness
- Utilize the BW in full
- Equilibrium point shifts right
- Reason: BBRv1's aggressive startup, no response to ReTx

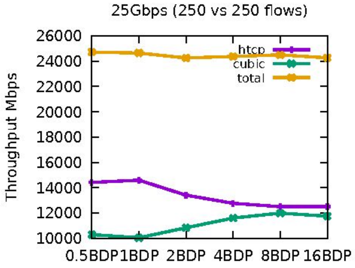


BBR v2 vs CUBIC

- Reduced BDP dependency for low BW
- BDP dependency in fairness remains
- Utilize the BW in full
- Equilibrium point shifts right

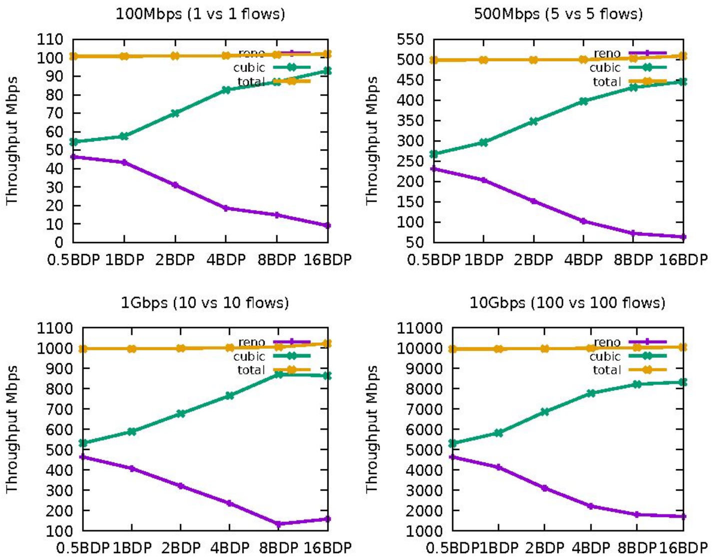
Observation on CCAs when AQM is FIFO (2/2)

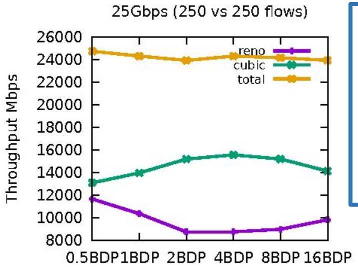




H-TCP vs CUBIC

- Throughput drops lightly with BDP
- Utilize the BW in full
- Comparatively low performance in high BWs due to estimation problem

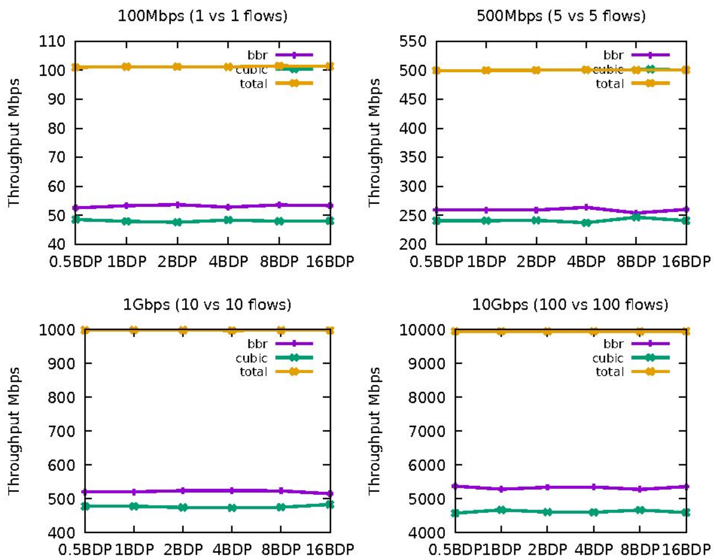


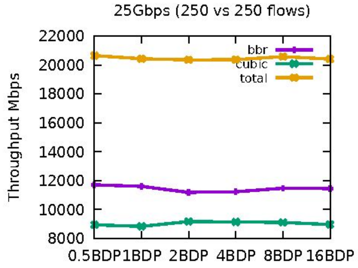


Reno vs CUBIC

- Decrease in throughput with increase in BDP
- Utilize the BW in full

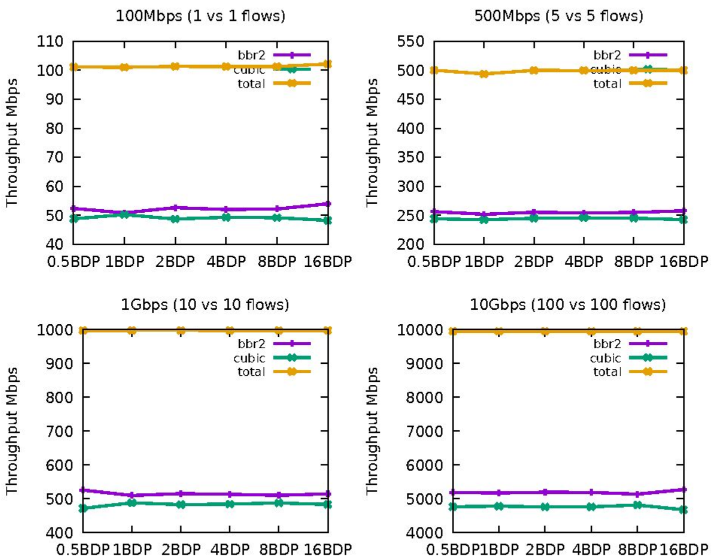
Observation on CCAs when AQM is Fq_codel (1/2)

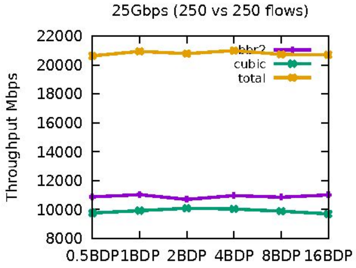




BBR v1 vs CUBIC

- Fq_codel removes the dependency on BDP
- Could not utilize the full 25 Gbps BW

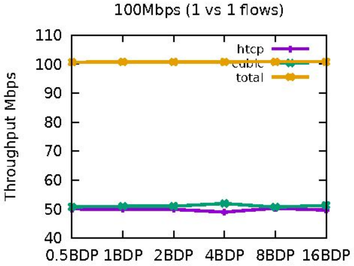


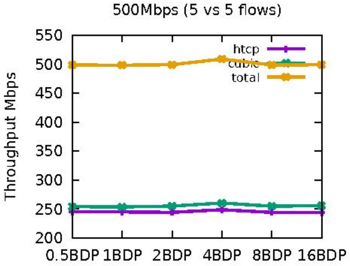


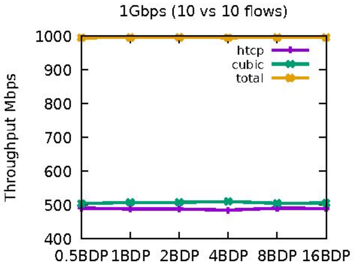
BBR v2 vs CUBIC

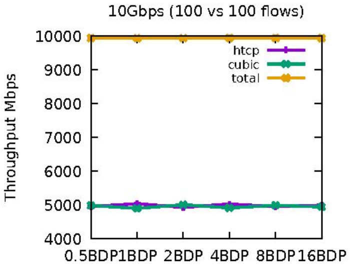
- Fq_codel removes the dependency on BDP
- Could not utilize the full 25 Gbps BW

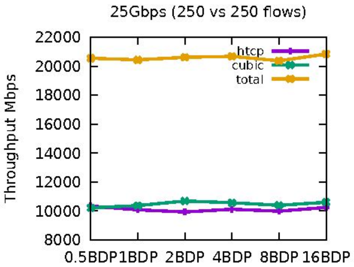
Observation on CCAs when AQM is Fq_codel (2/2)





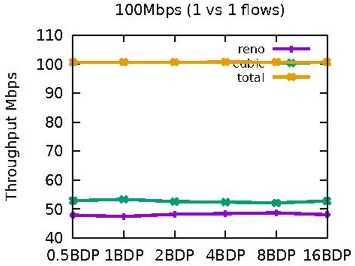


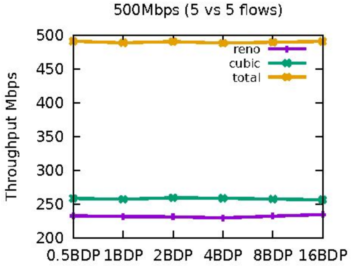


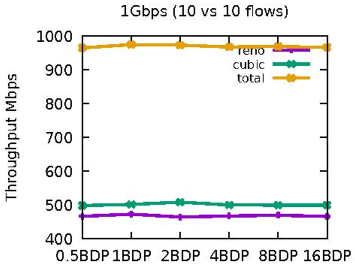


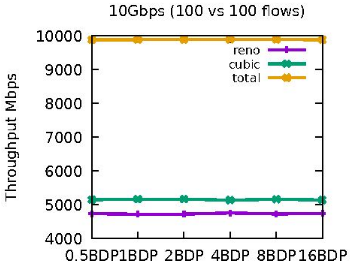
H-TCP vs CUBIC

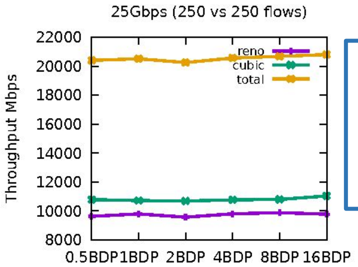
- No dependency on BDP
- Could not utilize the 25 Gbps BW in full







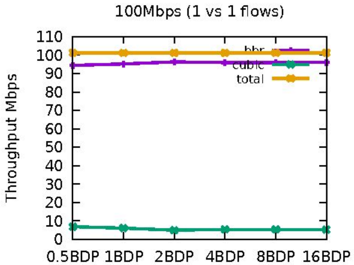


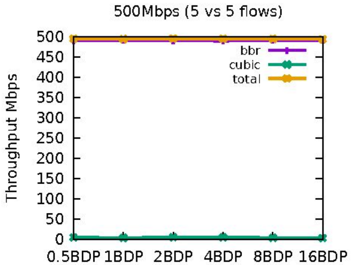


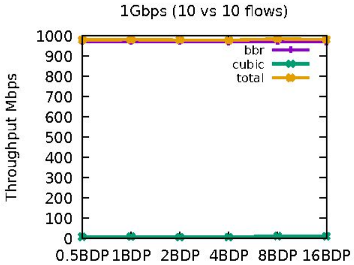
Reno vs CUBIC

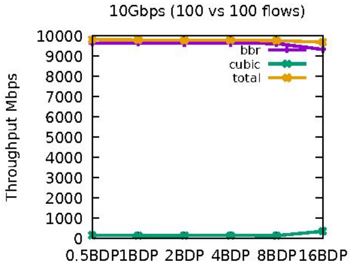
- No dependency on BDP
- Could not utilize the 25 Gbps BW in full

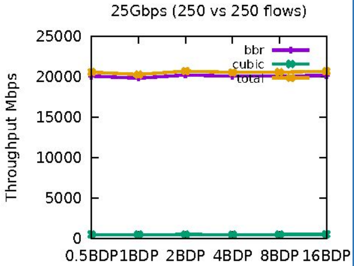
Observation on CCAs when AQM is RED (1/2)





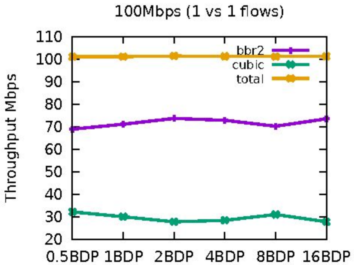


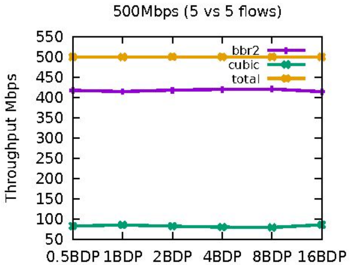


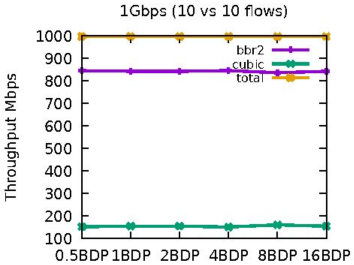


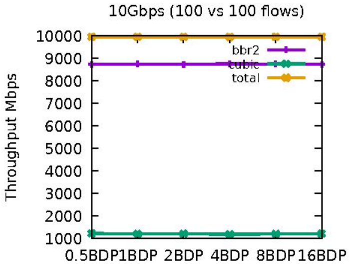
BBR v1 vs CUBIC

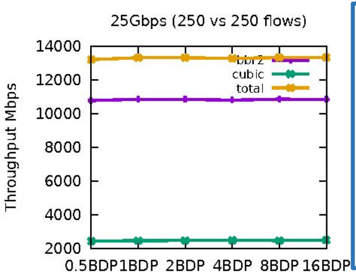
- No dependency on BDP
- Fairness is very poor, BBR v1 chocks up all the BW
- Could not utilize the 25 Gbps BW in ful
- Reason: RED's packet drop probability - based on queue length/waiting time







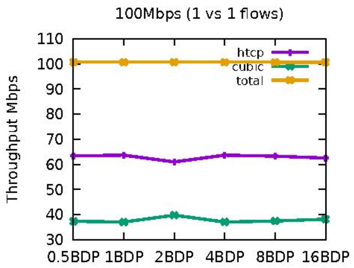


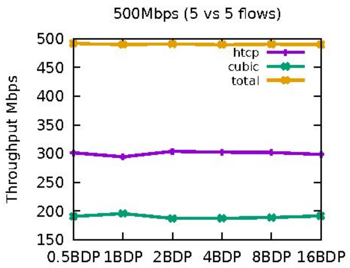


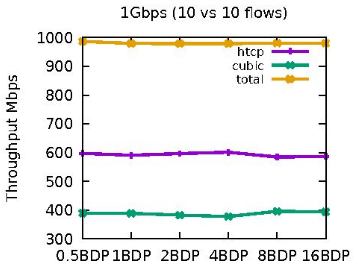
BBR v2 vs CUBIC

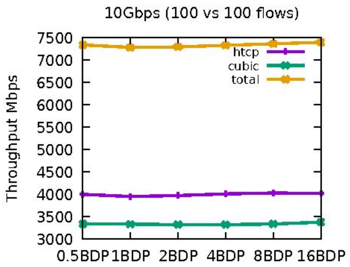
- No dependency on BDP
- Small improvement by BBR v2 in fairness than BBR v1, but still chocks up all the BW.
- Could only have a throughput of 1.4 Gbps in the 25 Gbps link.

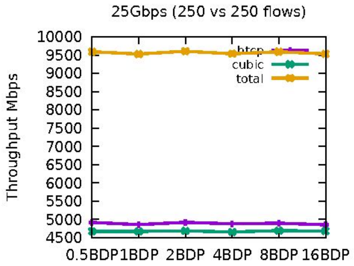
Observation on CCAs when AQM is RED (2/2)





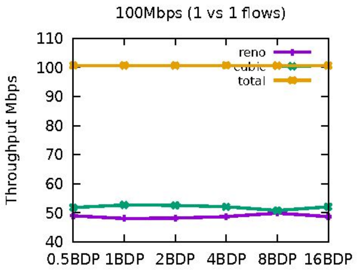


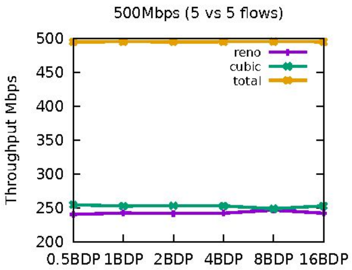


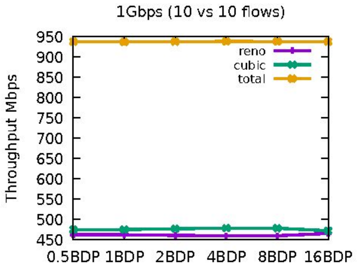


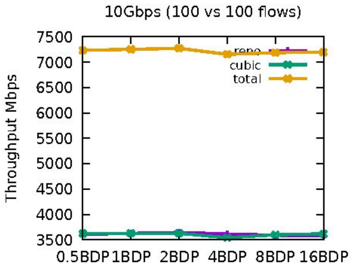
H-TCP vs CUBIC

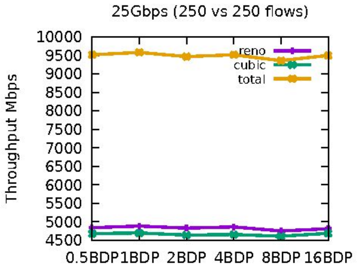
- No dependency on BDP
- Significantly low fairness
- Could not utilize the 10 and 25 Gbps BWs







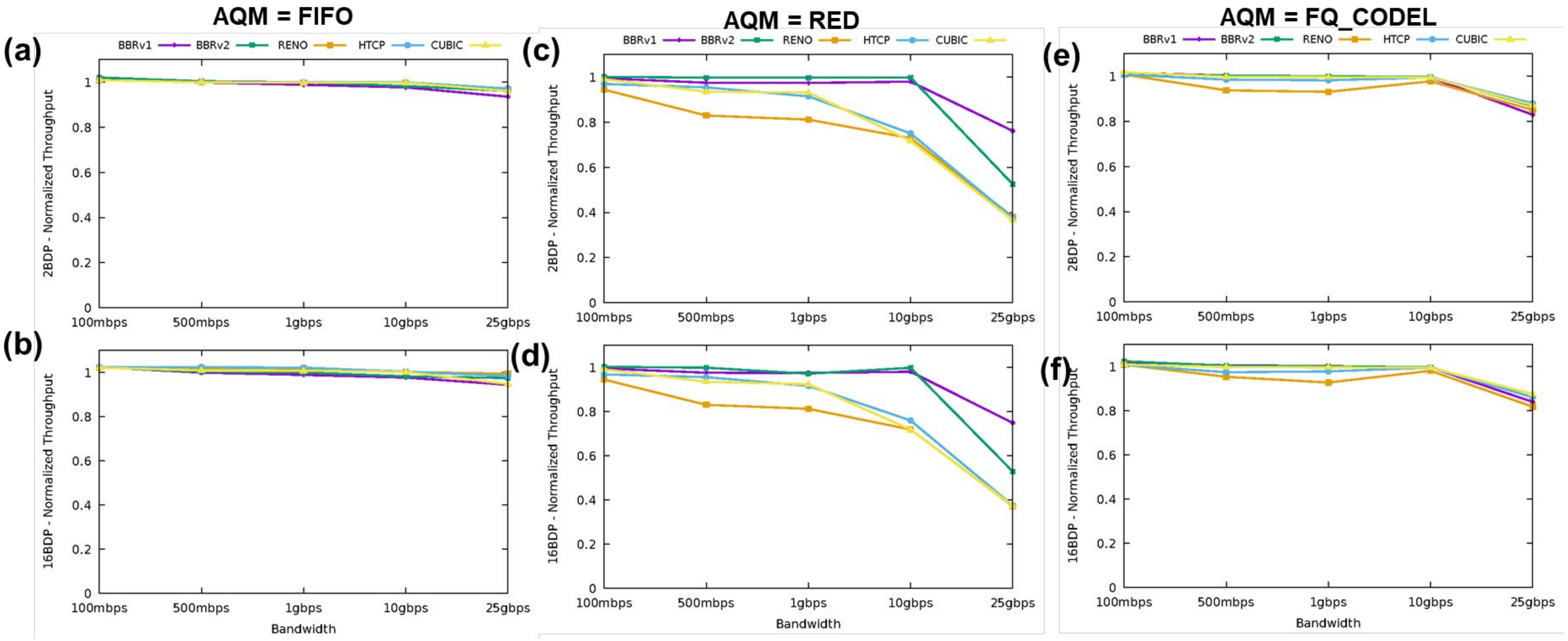




Reno vs CUBIC

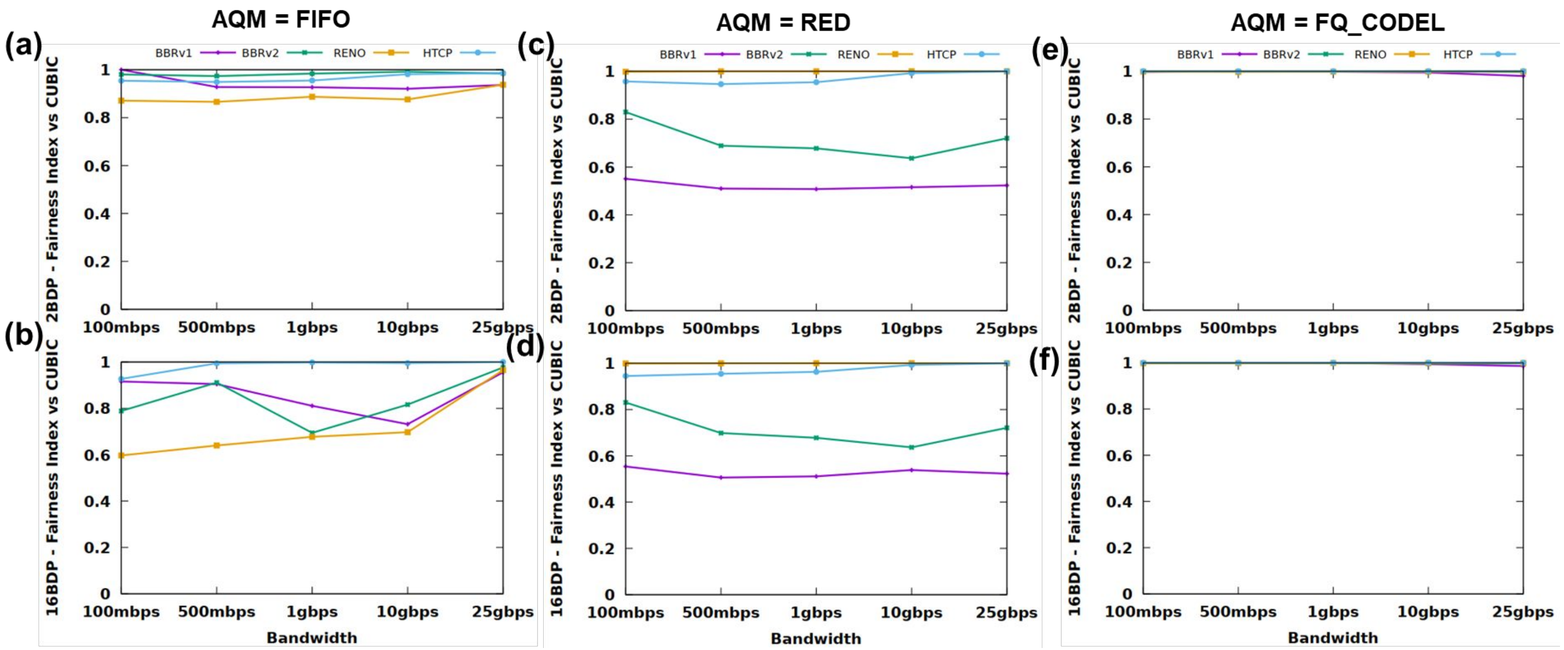
- No dependency on BDP
- Better fairness than others
- Could not utilize the 10 and 25 Gbps BWs

Overall Link Utilization - intra CCAs



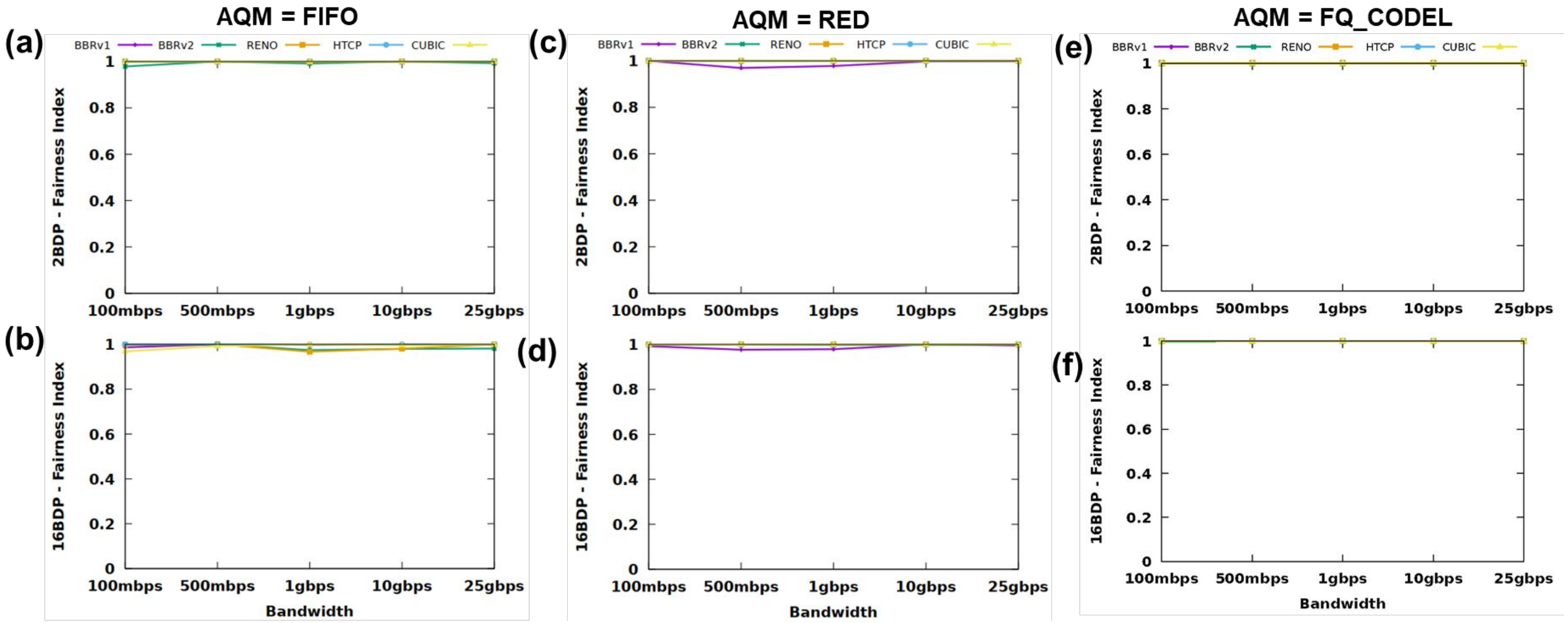
Point-to-point interplay – During intra-CCA experiments, observed overall link utilization for: (a) – (b) FIFO, (c) – (d) RED, and (e) – (f) FQ CODEL.

Jain’s Fairness Index - inter CCAs



Point-to-point interplay – During inter-CCA experiments, observed Jain's fairness index – vs CUBIC: (a) – (b) FIFO, (c) – (d) RED, and (e) – (f) FQ CODEL.

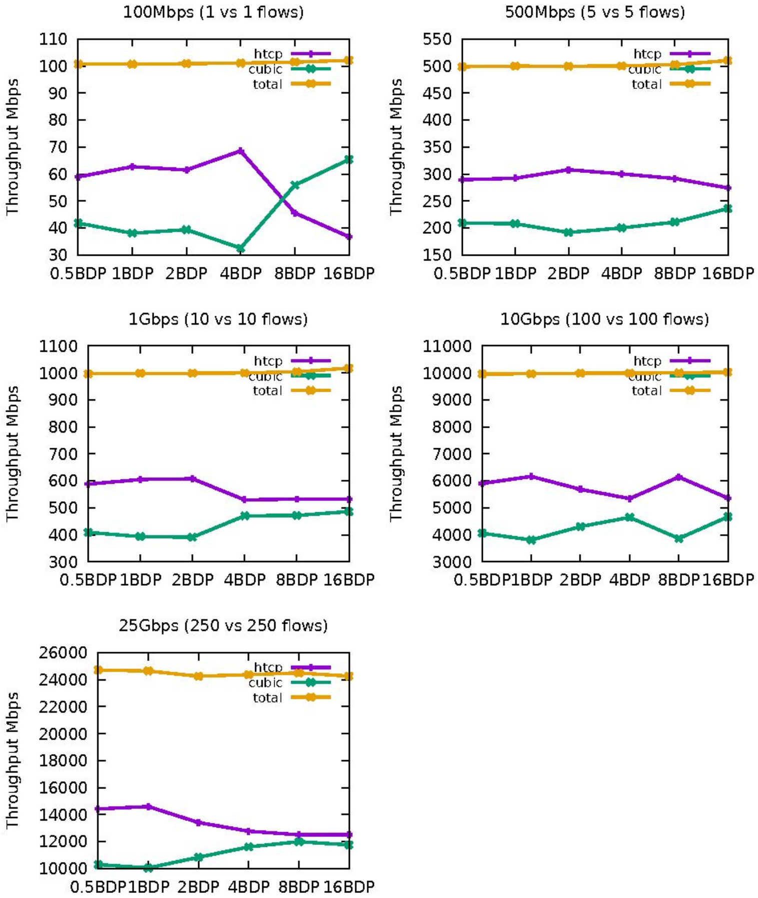
Jain's Fairness Index - intra CCAs



Point-to-point interplay – During inter-CCA experiments, observed Jain's fairness index – vs CUBIC: (a) – (b) FIFO, (c) – (d) RED, and (e) – (f) FQ CODEL.

Conclusion

- The choice of queuing algorithm plays a significant role in achieving high fairness for flows.
- Google's BBRv2 transfer protocol shows better performance in terms of retransmissions than H-TCP, but lacks fairness when competing with CUBIC flows. **We use CUBIC and H-TCP in DOE.**
- FQ_CODEL – superior fairness, fails to use full capacity 25 Gbps
  - **Solution** – research on fixing internal parameters.
- Conclusion: combining Fq_codel queuing method with BBR v2 may offer the best balance of low retransmissions, high fairness, and good bandwidth utilization across a range of scenarios.



GitHub Repo:
https://github.com/poseidon-workflows/tcp-conflict-study

Acknowledgements

U.S. DEPARTMENT OF ENERGY

DOE ASCR Award (DE-SC0022328): Integrated Computational and Data Infrastructure (ICDI) Program

NSF

FABRIC