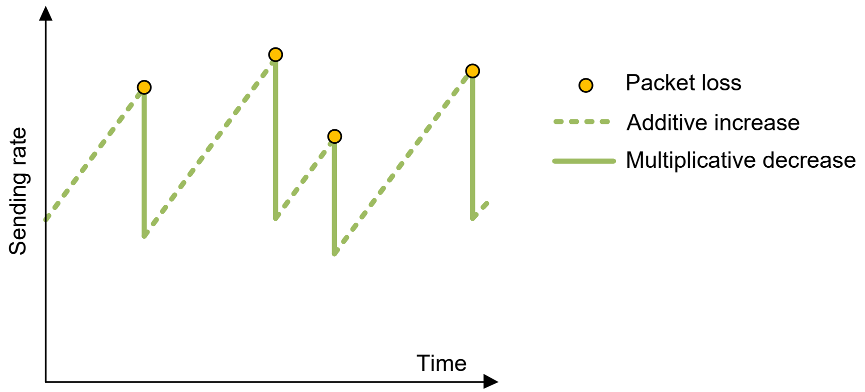
P4CCI: P4-based Online TCP Congestion Control Algorithm Identification for Traffic Separation

Elie Kfoury*, Jorge Crichigno* (Presenter), Elias Bou-Harb^
*College of Engineering and Computing, University of South Carolina
^Cyber Center For Security and Analytics, University of Texas at San Antonio, USA

IEEE International Conference on Communications
May 30, 2023
Rome, Italy

TCP Traditional Congestion Control

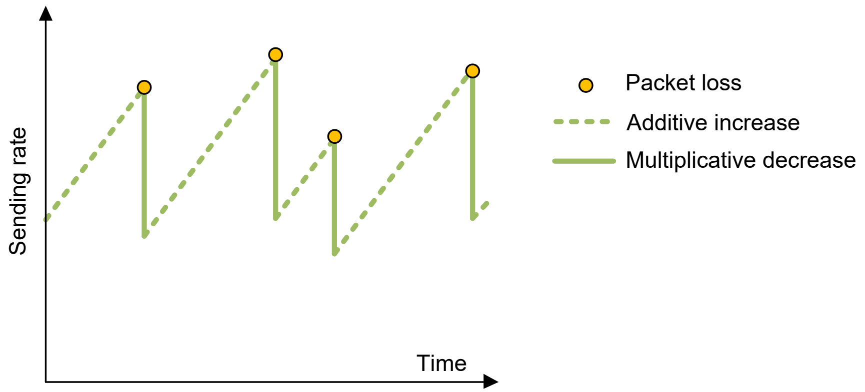
- The principles of window-based CC were described in the 1980s[1]
- Traditional CC algorithms follow the additive-increase multiplicative-decrease (AIMD) form of congestion control

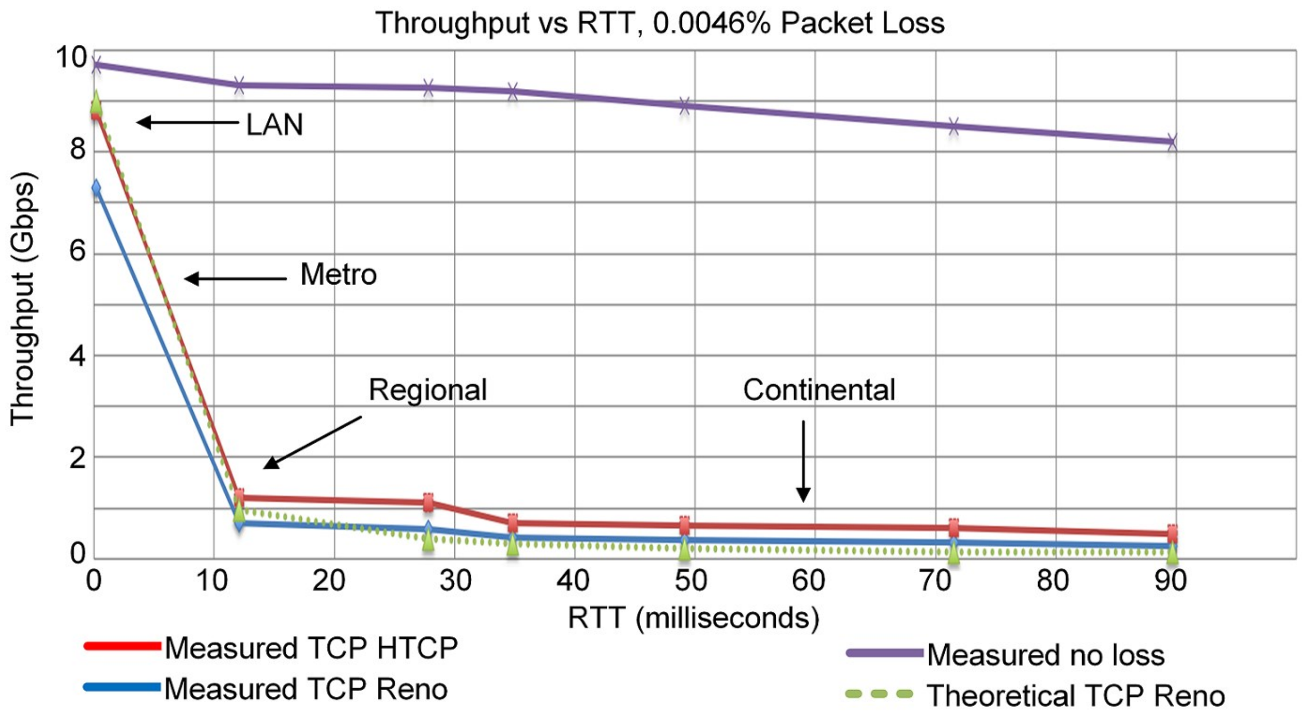


1. V. Jacobson, M. Karels, Congestion avoidance and control, ACM SIGCOMM Computer Communication Review 18 (4) (1988).

TCP Traditional Congestion Control

- The principles of window-based CC were described in the 1980s[1]
- Traditional CC algorithms follow the additive-increase multiplicative-decrease (AIMD) form of congestion control

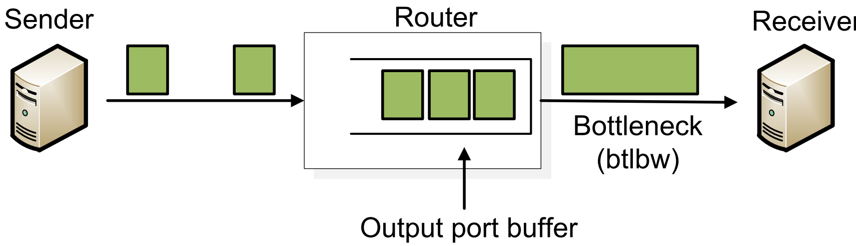


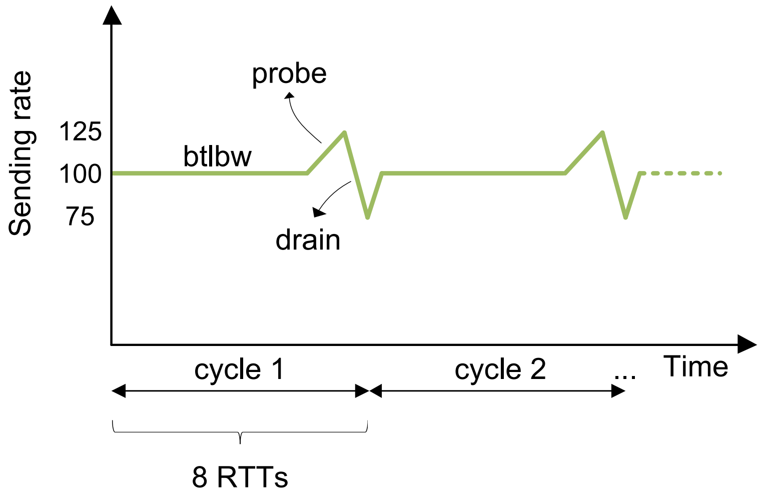


1. V. Jacobson, M. Karels, Congestion avoidance and control, ACM SIGCOMM Computer Communication Review 18 (4) (1988).

BBR: Model-based CC

- TCP Bottleneck Bandwidth and RTT (BBR) is a rate-based congestion-control algorithm[1]
- BBR represented a disruption to the traditional CC algorithms:
  - is not governed by AIMD control law
  - does not the use packet loss as a signal of congestion
- At any time, a TCP connection has one slowest link bottleneck bandwidth (btlbw)

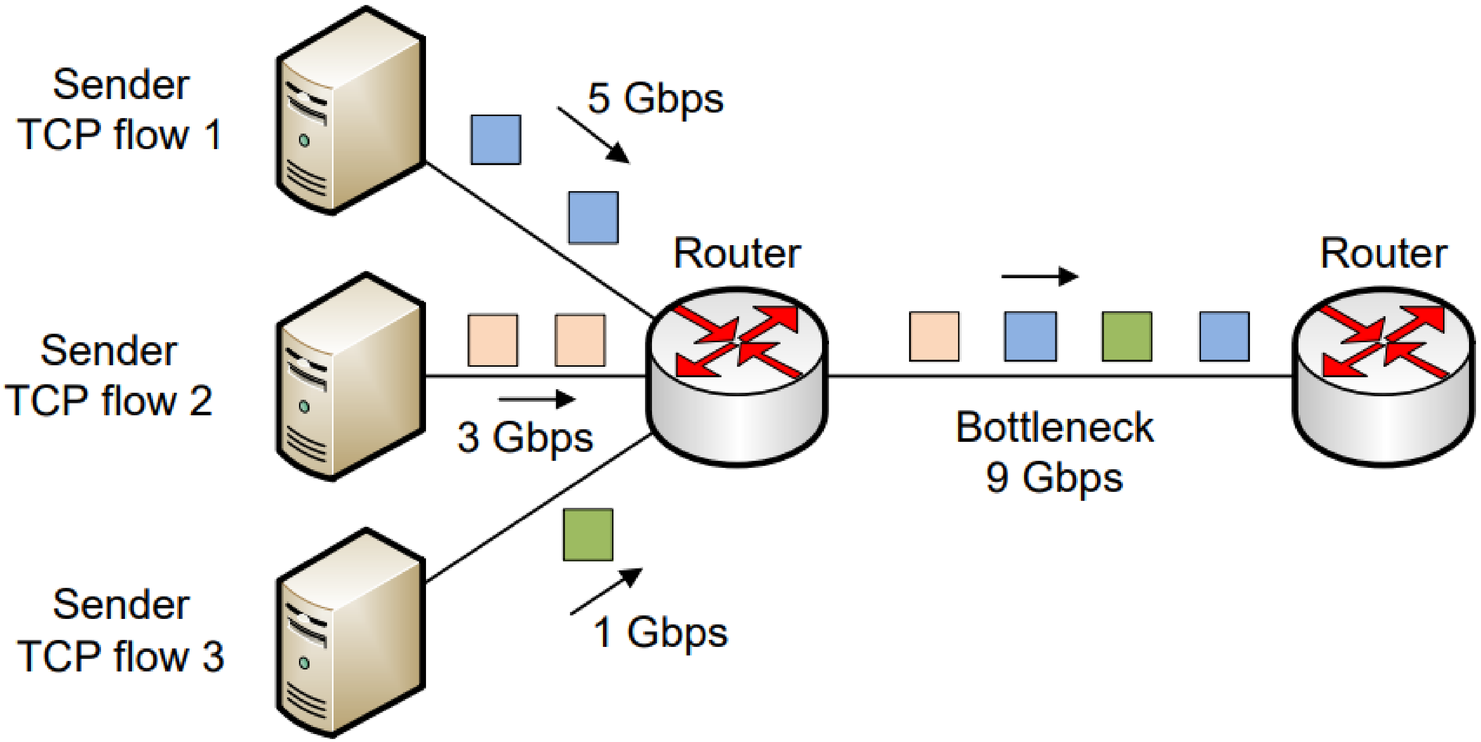




1. N. Cardwell et al. "BBR v2, A Model-based Congestion Control." IETF 104, March 2019.

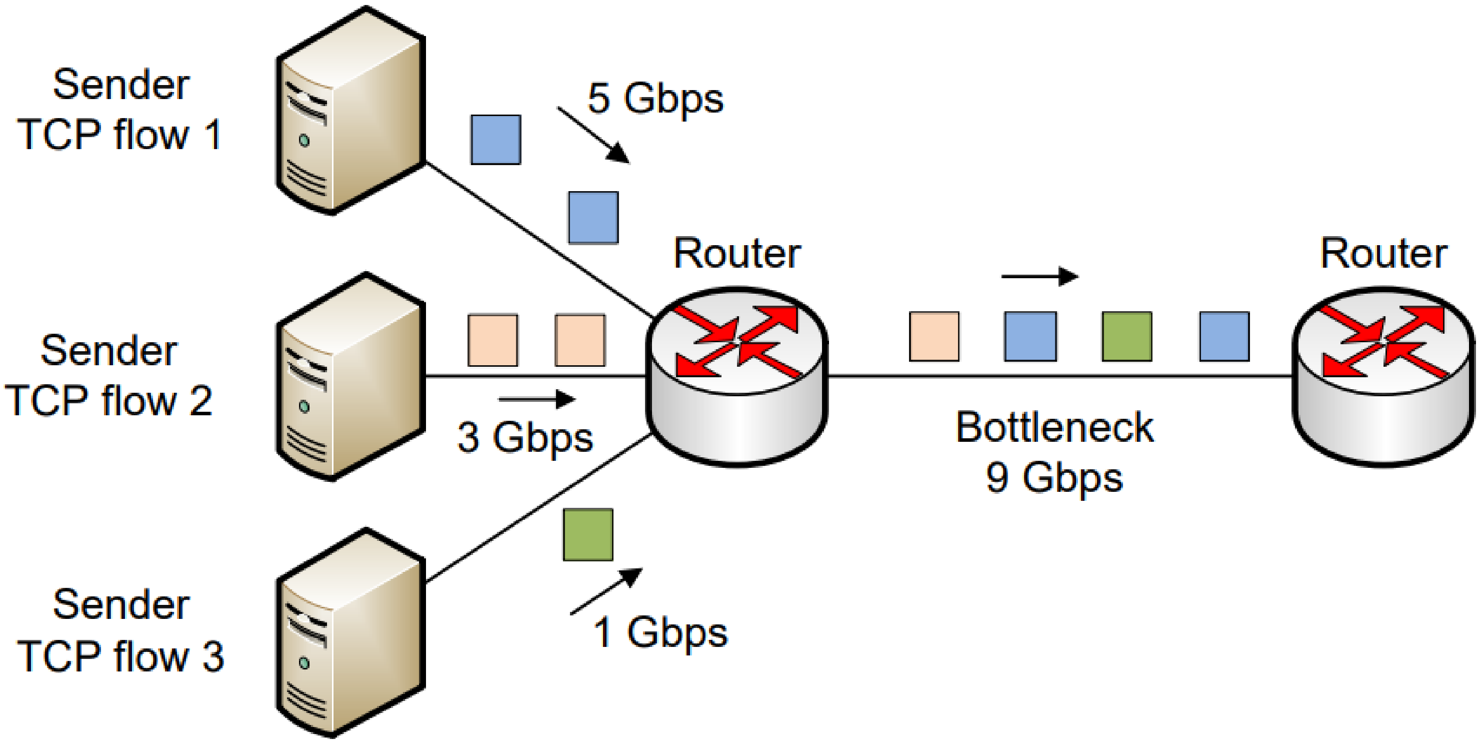
Fairness

- Fairness: how fair is the capacity of the link being divided among the competing flows
- Jain's fairness index: $I = \frac{\left(\sum_{i=1}^{n} T_i\right)^2}{n \sum_{i=1}^{n} T_i^{\,2}}$



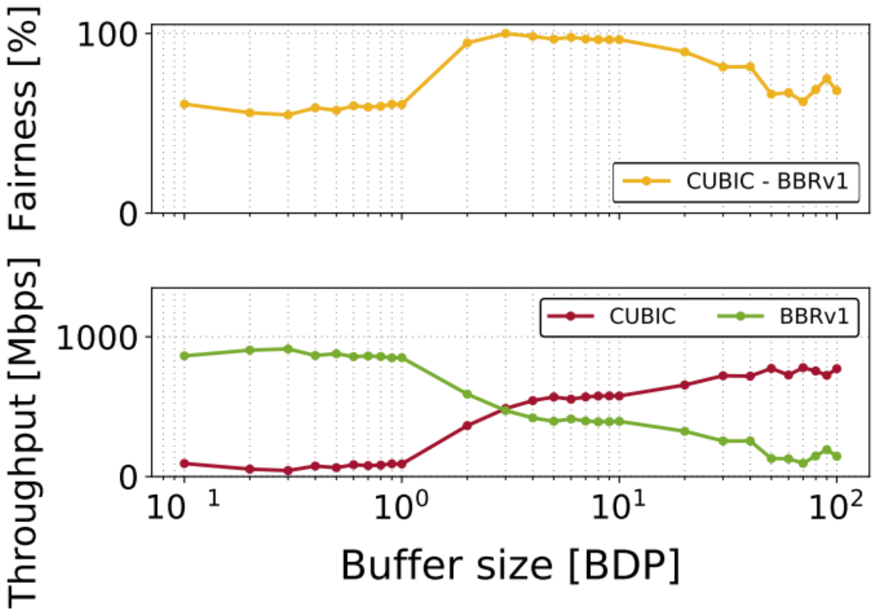
Fairness

- Fairness: how fair is the capacity of the link being divided among the competing flows
- Jain's fairness index: $I = \frac{(\sum_{i=1}^{3} T_i)^2}{3\sum_{i=1}^{3} T_i^{\,2}} = \frac{(5\cdot 10^9 + 3\cdot 10^9 + 1\cdot 10^9)^2}{3\cdot((5\cdot 10^9)^2 + (3\cdot 10^9)^2 + (1\cdot 10^9)^2)} = \boxed{0.77}$

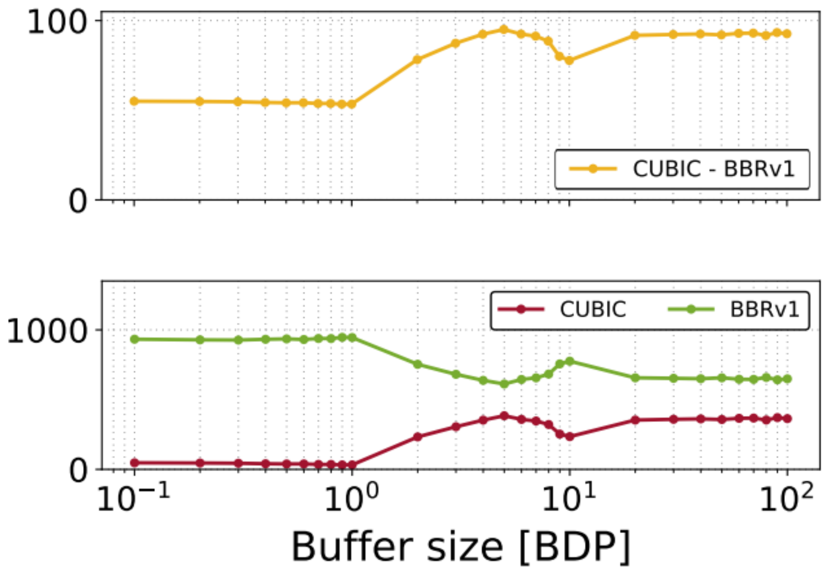


Fairness

- The fairness between flows belonging to different CCAs is often low
- E.g., the fairness among Cubic and BBR flows[1]



(1 CUBIC, 1 BBR)

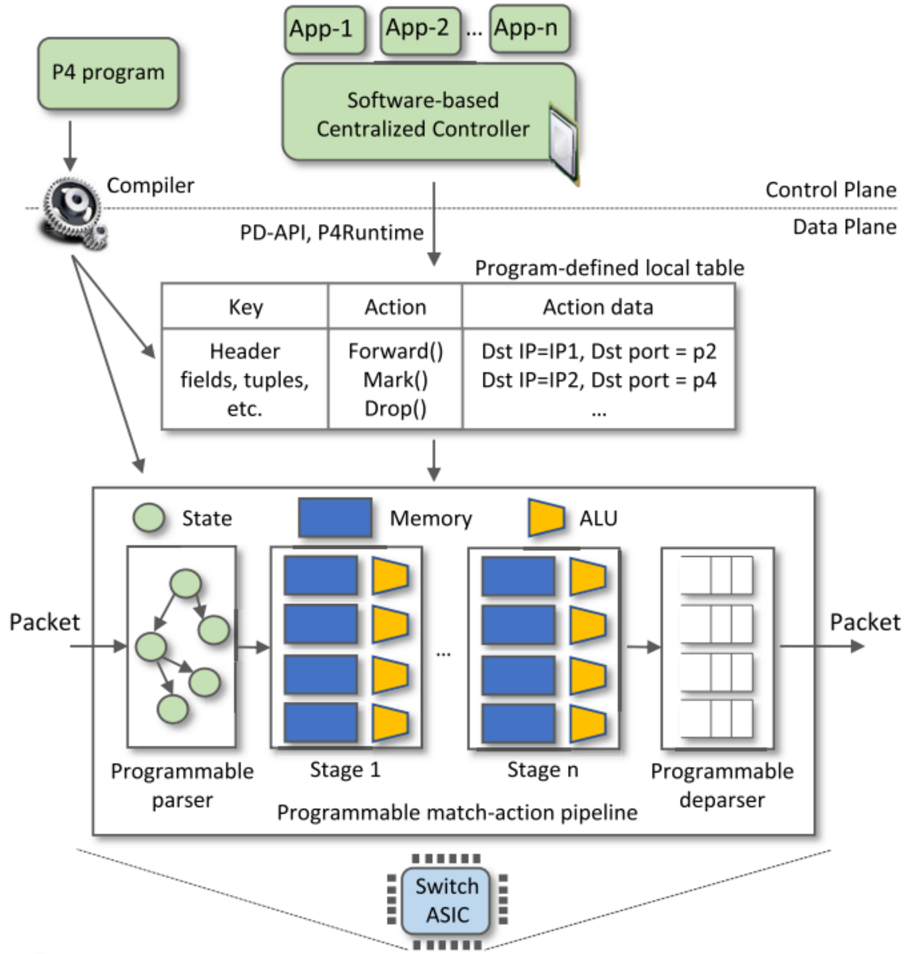


(50 CUBIC, 50 BBR)

1. E. Kfoury, J. Gomez, J. Crichigno, E. Bou-Harb, "An Emulation-based Evaluation of TCP BBRv2 Alpha for Wired Broadband", Computer Communications, July 2020.

P4 Programmable Data Planes

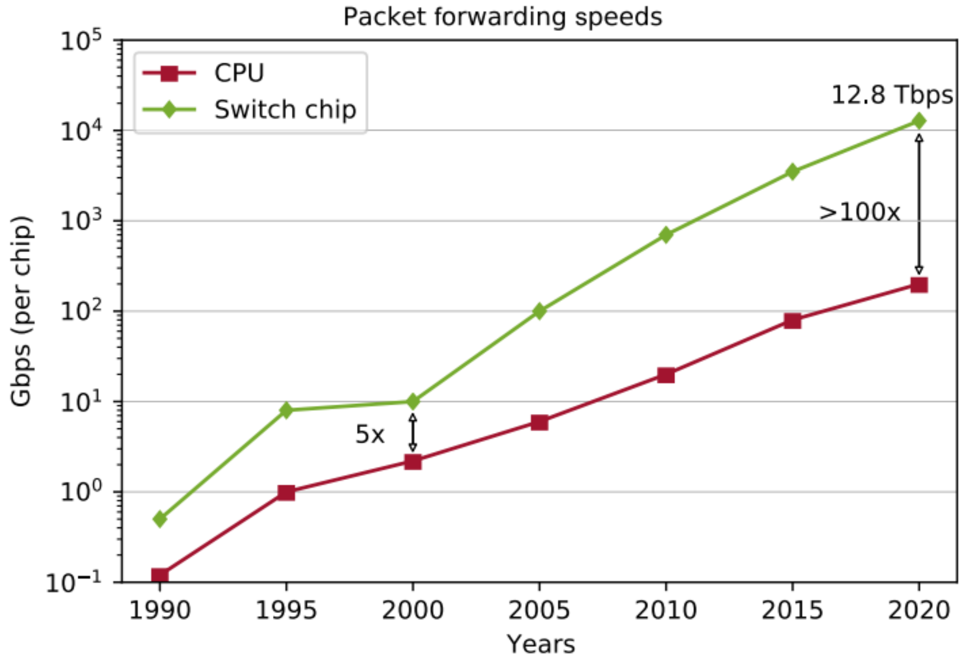
- P4[1] Programmable Data Planes (PDPs) permit a programmer to program the data plane
  - ➢ Define and parse new protocols
  - ➢ Customize packet processing functions
  - ➢ Measure events occurring in the data plane with high precision
  - ➢ Offload applications to the data plane



1. P4 stands for stands for Programming Protocol-independent Packet Processors

P4 Programmable Data Planes

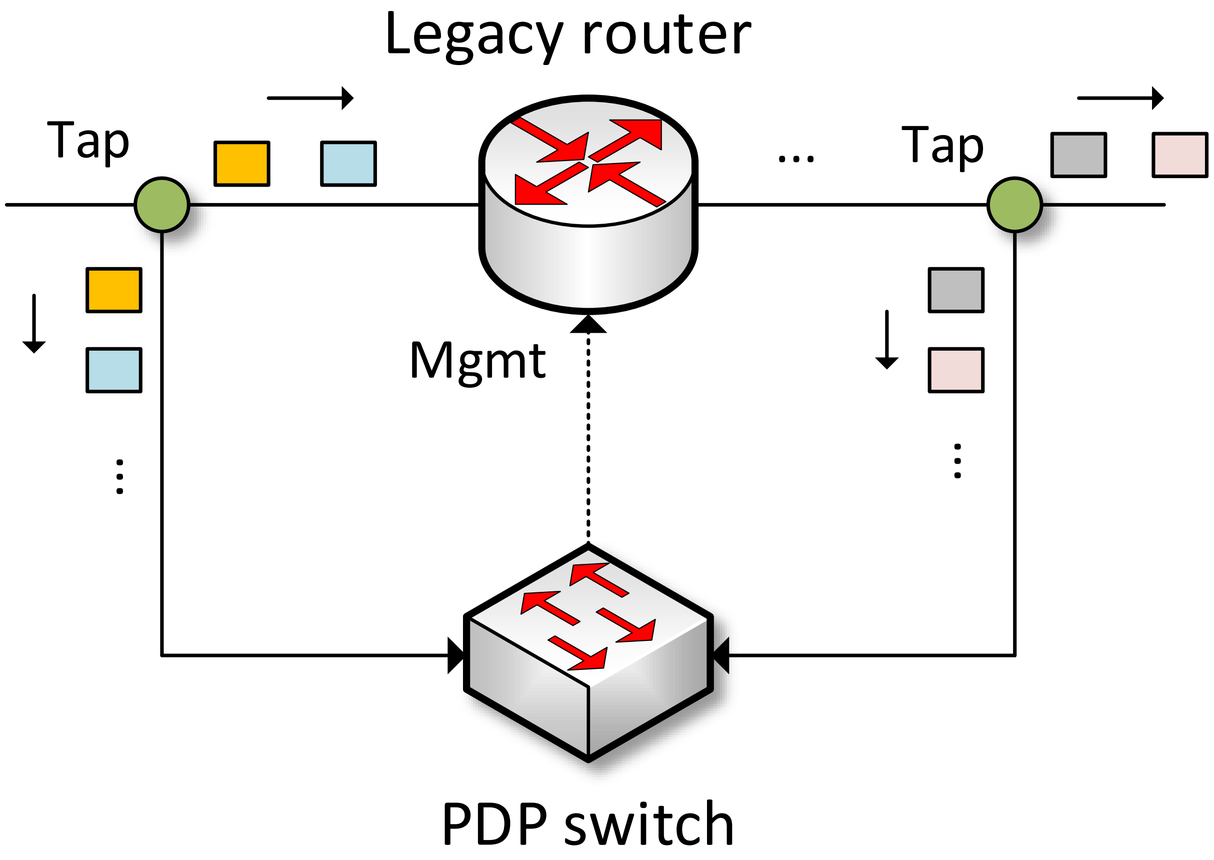
- P4[1] Programmable Data Planes (PDP) permit a programmer to program the data plane
  - Define and parse new protocols
  - Customize packet processing functions
  - Measure events occurring in the data plane with high precision
  - Offload applications to the data plane
  - **If the P4 program compiles, it runs on the chip at line rate**



Reproduced from N. McKeown. Creating an End-to-End Programming Model for Packet Forwarding. Available: **https://www.youtube.com/watch?v=fiBuao6YZI0&t=4216s**

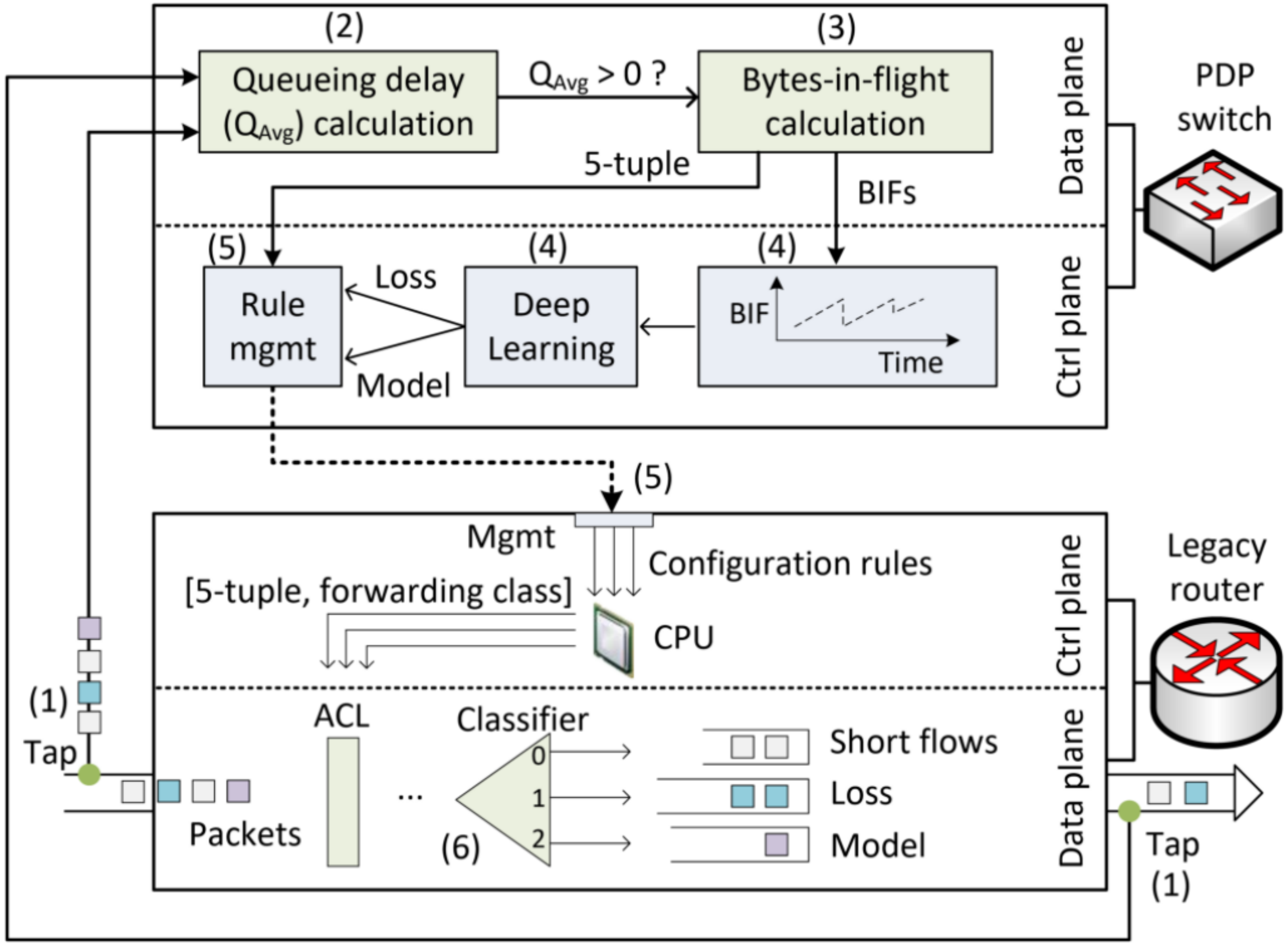
Proposed System

- Passive PDPs for congestion control algorithm (CCA) identification at line rate
- The PDP measures the average queueing
- During congestion, the PDP computes the flow's bytes-in-flight (BIF)
- Deep learning model classifies the CCA using the flow's BIF values
- Flows belonging to the same CCA are assigned to dedicated queues.



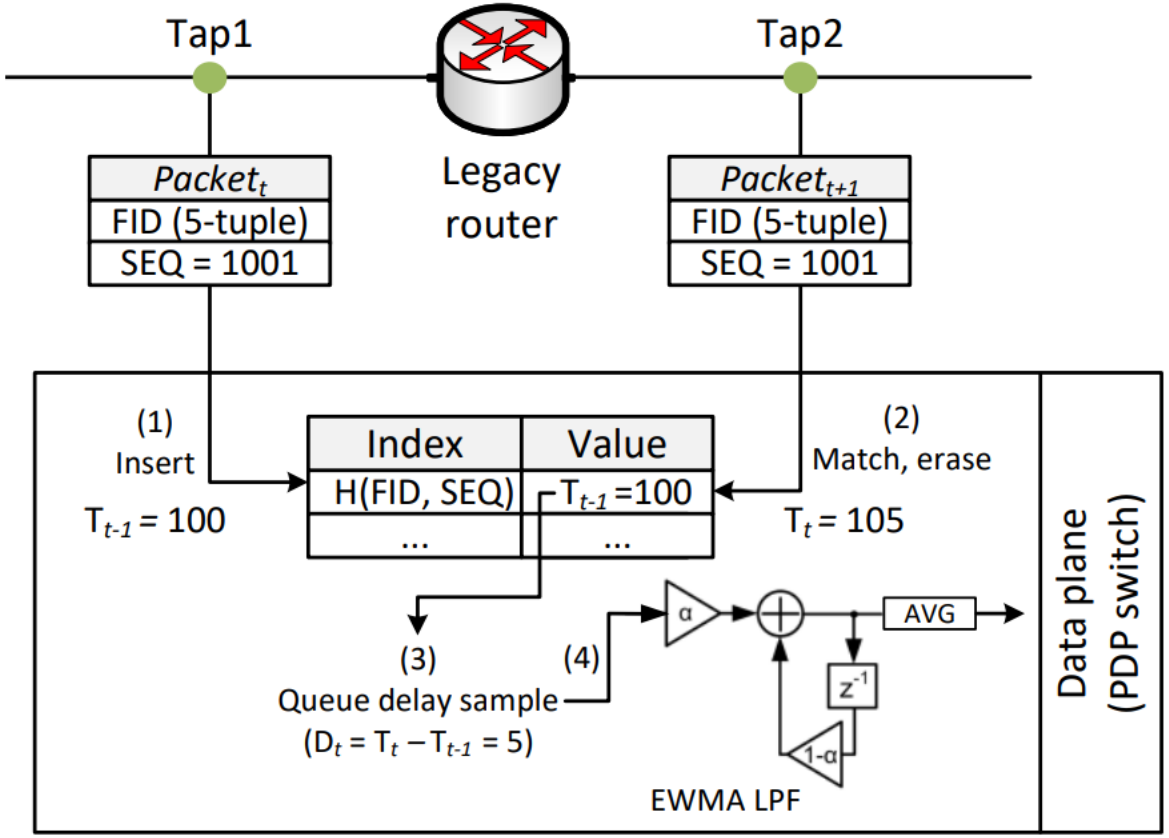
Proposed System

- Passive PDPs for congestion control algorithm (CCA) identification at line rate
- The PDP measures the average queueing
- During congestion, the PDP computes the flow's bytes-in-flight (BIF)
- Deep learning model classifies the CCA using the flow's BIF values
- Flows belonging to the same CCA are assigned to dedicated queues.



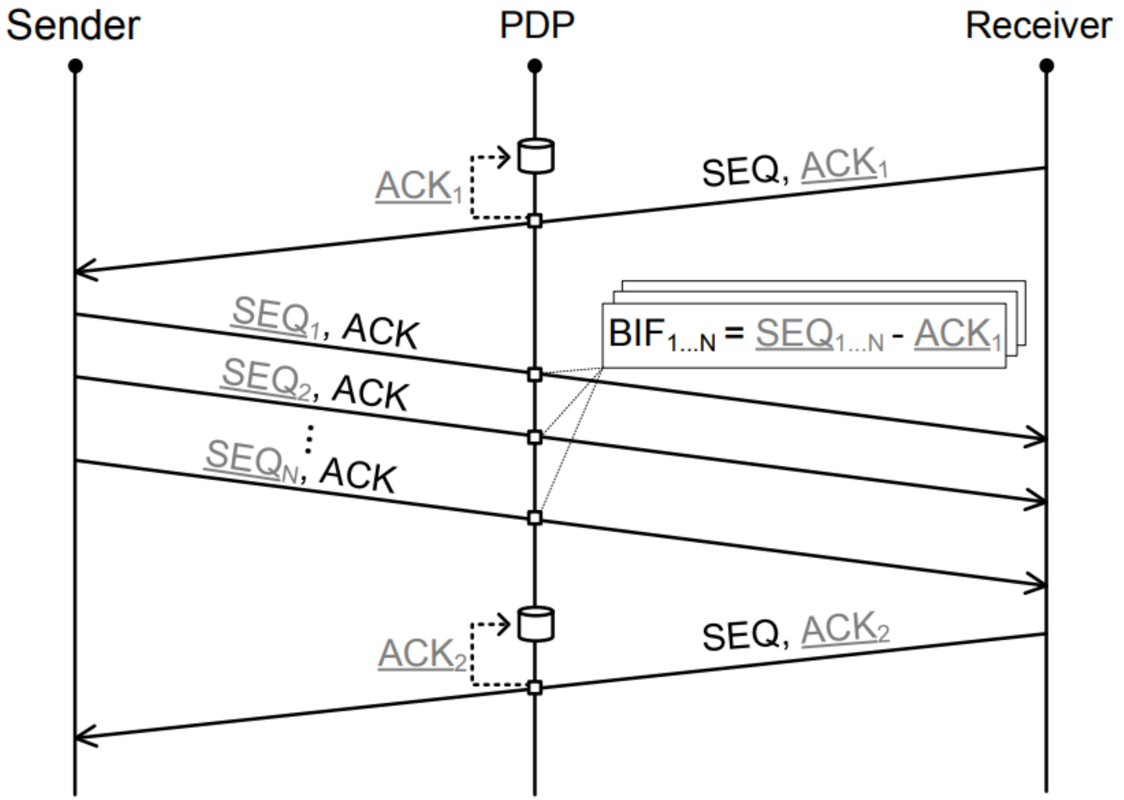
Queue Delay Calculation

- The queueing delay is calculated by leveraging the precise timestamp of the hardware switch (nanosecond resolution)
- The queueing delay sample is fed to an Exponentially Weighted Moving Average (EWMA)



Bytes-in-flight Calculation

- Bytes-in-flight (BIF) is the amount of data sent but not yet acknowledged
- BIF is correlated to the TCP congestion window

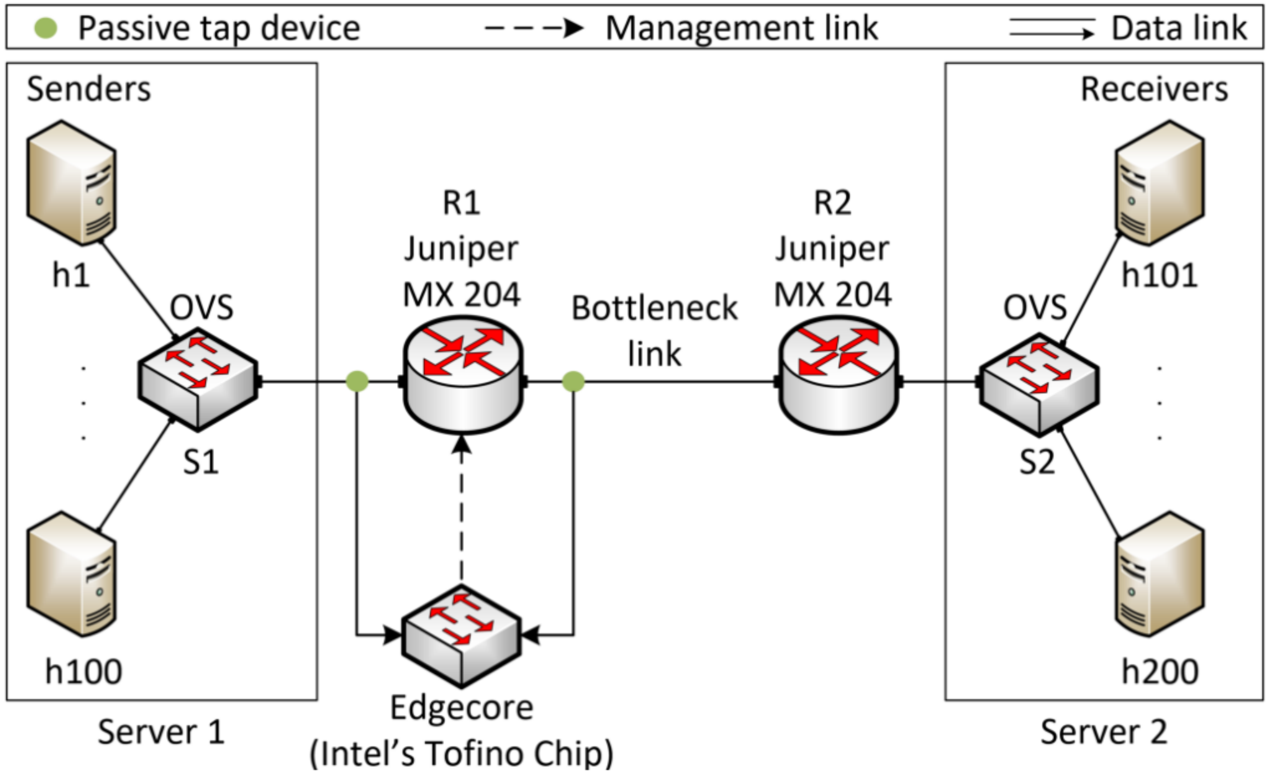


Time Series Preparation and Deep Learning

- BIF values are pushed to the control plane of the PDP switch during congestion
- A time series is constructed
- Two pre-processing steps:
  - **Outliers Rejection**: z-score method, which uses the MAD (Median Absolute Deviation), is used
  - **Normalization**: The time series is preprocessed using z-normalization
- Fully Convolutional Neural Networks (FCNs) used to classify the univariate time series (deep learning)

Experimental Topology

- Mininet was used to emulate the hosts running in network namespaces in Linux
- The senders are connected to a virtual switch (Open vSwitch)
- The server's interface is connected to a Juniper router (MX-204)
- The PDP device is Intel's Tofino programmable ASIC that operates at 3.2 Tbps



Model Training

- The model is trained on CAIDA's dataset
- The model is also trained with synthetically generated traffic

TRAINING PARAMETERS FOR THE SYNTHETICALLY GENERATED DATASET

Flows	1, 2, 5, 10, 15, 20, 50, 100
Bandwidth [bps]	500M, 1G, 2G, 3G, 4G, 5G, 10G
CCAs	Loss (CUBIC, Reno), Model (BBR)
Packet loss rates [%]	0, 0.1, 0.25, 0.5
Propagation delays [ms]	0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100
Buffer sizes [ms]	10, 20, 30, 40, 50, 60, 70, 80, 90, 100

Model Testing

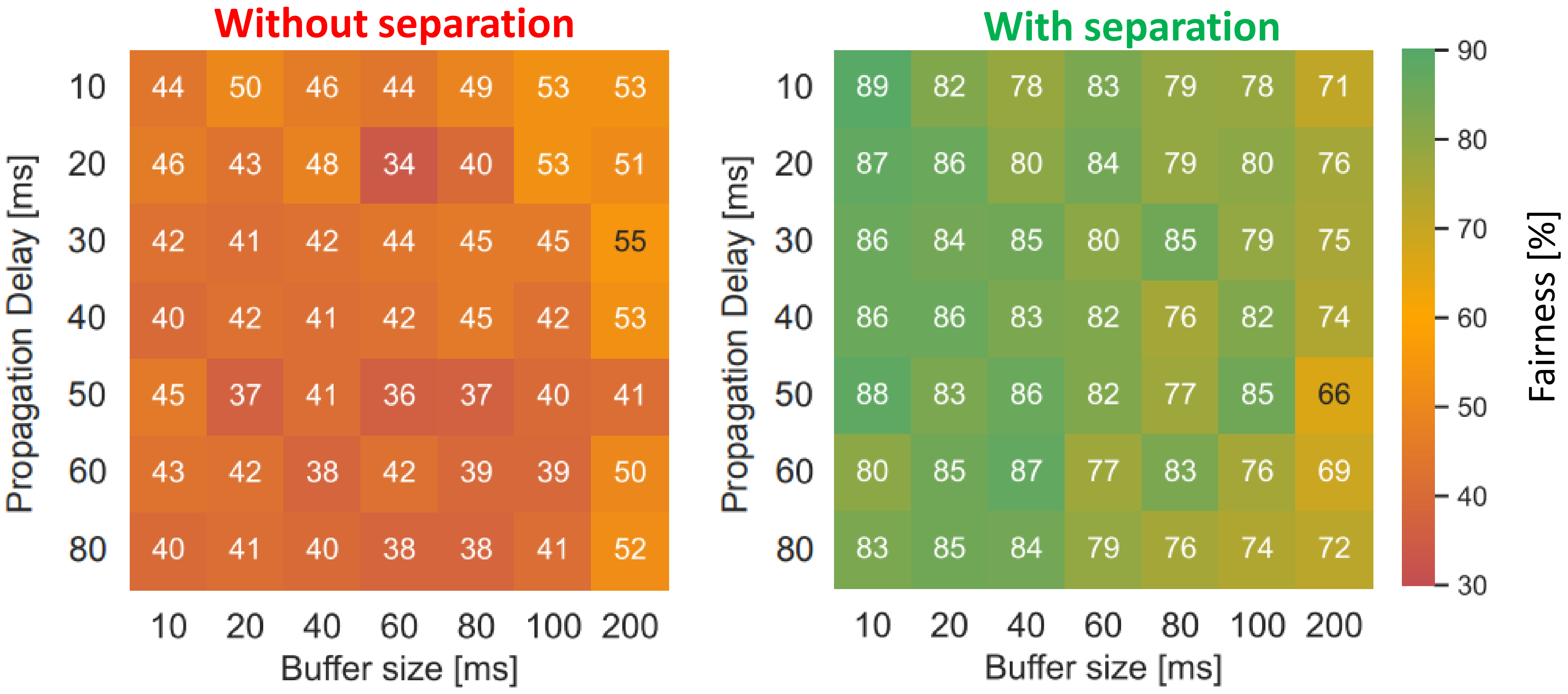
- The model was tested against 10 minutes of traffic from the remaining CAIDA dataset
- The bottleneck bandwidth was configured to 1Gbps, 1.5Gbps, 2Gbps, and 2.5Gbps
- Results outperformed the state-of-the-art CCA identification systems

CLASSIFICATION RESULTS.

Dataset	Classes	Precision	Recall	F1-score	Accuracy
CAIDA 1Gbps	Loss	96.2%	93.5%	94.8%	96.1%
	Model	96.0%	97.7%	96.8%	
CAIDA 1.5Gbps	Loss	95.2%	92.0%	93.1%	95%
	Model	95.6%	97.6%	96.6%	
CAIDA 2Gbps	Loss	92.0%	92.5%	92.3%	95.4%
	Model	96.9%	96.4	96.8%	
CAIDA 2.5Gbps	Loss	91.5%	91.0%	91.2%	95.6%
	Model	97.0%	97.1%	97.0%	
Synthetic	Loss	99.2%	99.5%	99.4%	99.4%
	Model	99.5%	99.2%	99.4%	

Fairness Evaluation

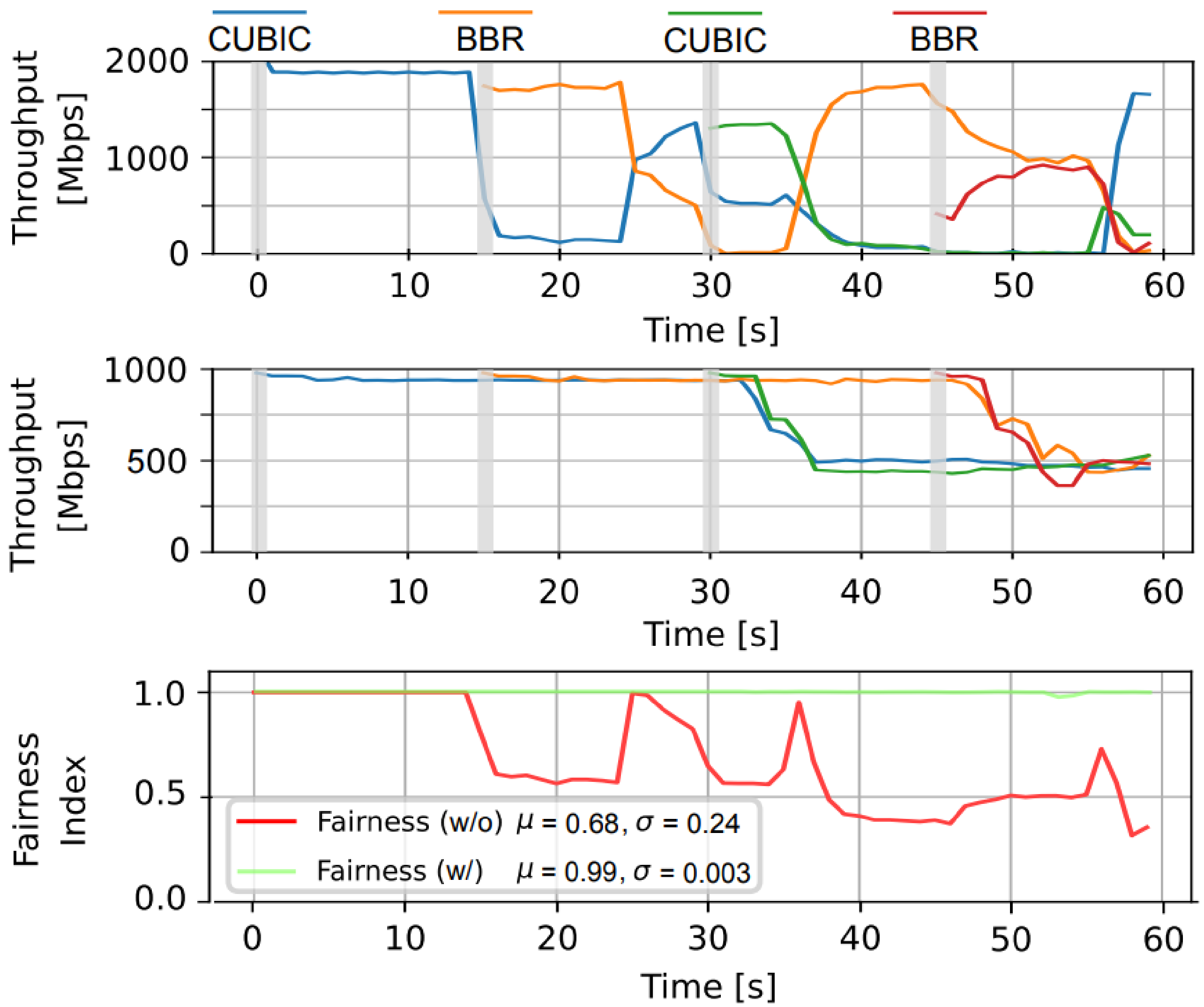
- 10 long flows (persistent over time) started within a few milliseconds of each other, with alternating CCAs
  - ➢ Flow1 uses CUBIC, Flow2 uses BBR, Flow3 uses CUBIC, etc.
- Various propagation delays and various router buffer sizes are used



(a) (b)

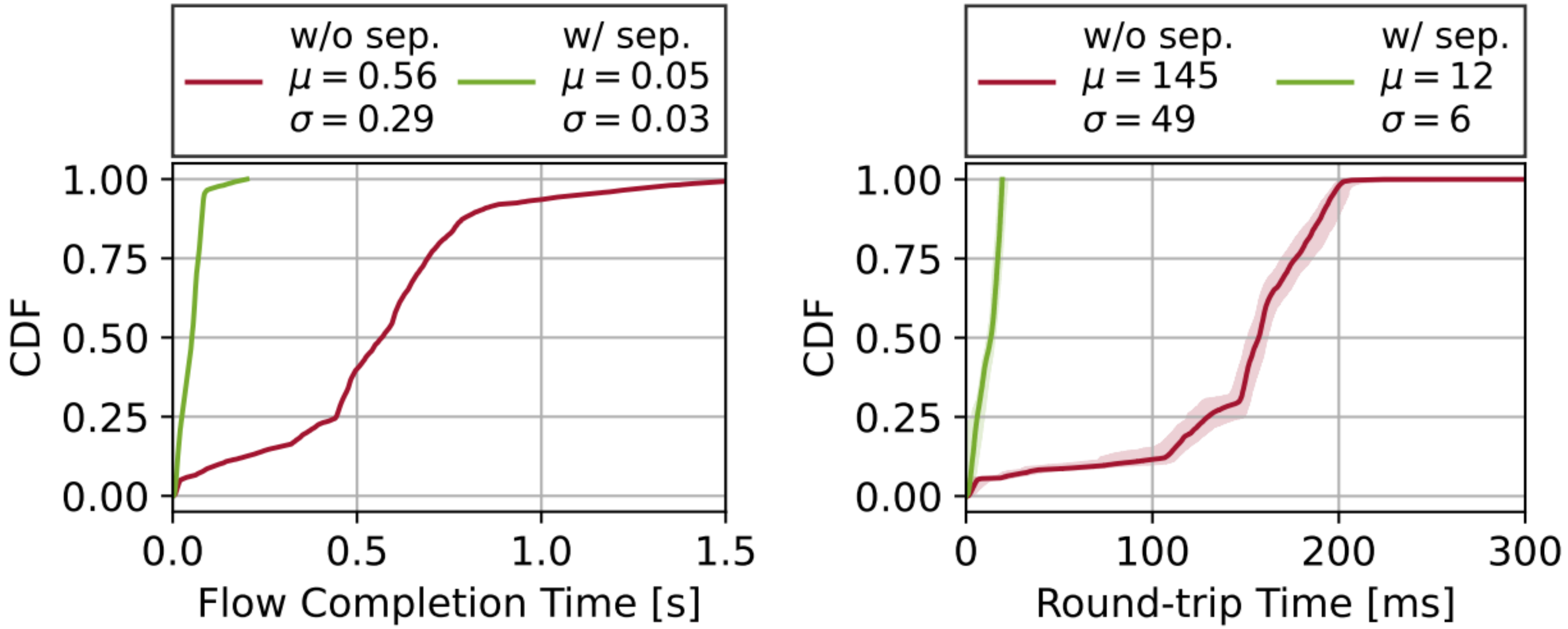
Fairness Evaluation

- Alternating flows joining every 15 seconds
- The system promptly identifies the CCA
- Fairness is ~ 100%



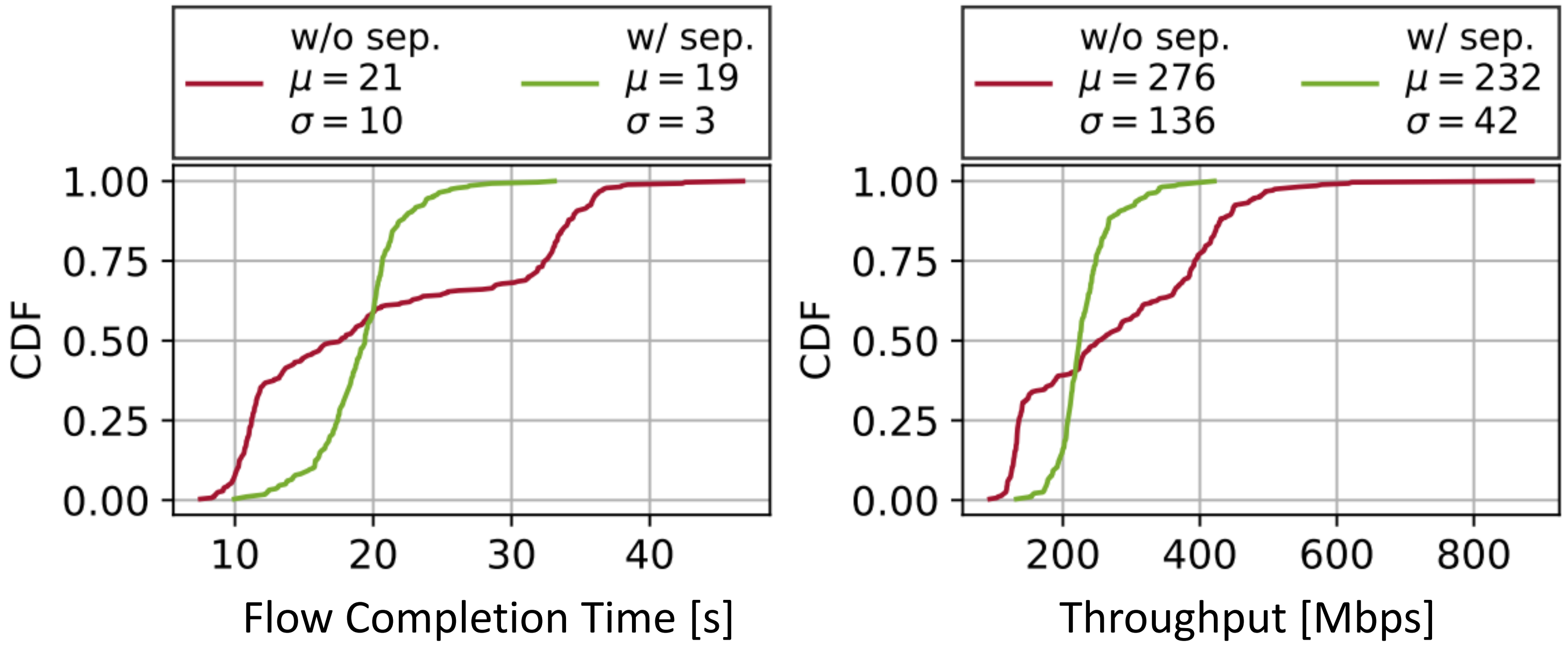
Flow Completion Time (Short Flows)

- 100 long flows (50% Cubic, 50% BBR) are generated over a bottleneck link of 3Gbps
- The queue size for the "w/o separation" scenario is 200ms
- 10,000 short flows, whose inter-connection times are generated from an exponential distribution with a mean of one second, are initiated



Flow Completion Time (Long Flows)

- 10 long flows started within few milliseconds of each other, with alternating CCAs
  - Flow1 uses CUBIC, Flow2 uses BBR, Flow3 uses CUBIC, etc.
- Each flow transfers a 500MB file
- In a fair network with a bottleneck of 2Gbps and 10 active flows:
  - Each flow is transferring at 200Mbps
  - FCT = 500MB / 200Mbps = 20s



Conclusion and Future Work

- This paper presented a system that uses passive PDPs to identify CCAs at line rate
- After identifying the CCA, the flow is enqueued into a dedicated queue based on the CCA variant
- The experiments were conducted on real hardware, and real datasets were used for testing
- One limitation is that the system assumes that the flows are uniformly distributed based on their CCA
- The authors plan to solve this queue assignment imbalance problem for future work

This work is supported by NSF award number 2118311

For additional information, please refer to
http://ce.sc.edu/cyberinfra/

Email: jcrichigno@cec.sc.edu, ekfoury@email.sc.edu

23